DM5804/DM6804
User's Manual

(Real Time Devices)

RTD Embedded Technologies Inc.

"Accessing the Analog World"®

BDM-610010019
Rev. A

DM5804/DM6804 User's Manual

RTD Embedded Technologies, INC.
103 Innovation Blvd.
State College, PA 16803-0906

Phone: +1-814-234-8087
FAX: +1-814-234-5218

E-mail
sales@rtd.com
techsupport@rtd.com

web site
http://www.rtd.com

Revision History

Rev. A	New manual naming method

Published by:

RTD Embedded Technologies, Inc.
103 Innovation Blvd.
State College, PA 16803-0906

Table of Contents

List of Illustrations

INTRODUCTION

The DM5804 dataModule™ timer/counter and digital I/O board turns your IBM PC-compatible cpuModule™ or other PC/104 computer into a high-performance timing, counting, and control system. Ultra-compact for embedded and portable applications, the DM5804 features:

- Five general purpose 16-bit timer/counters in an Am9513A chip,
- 24 timer/counter modes of operation,
- Binary or BCD up or down counting,
- On-board 5 MHz crystal,
- 24 TTL/CMOS 8255-based digital I/O lines which can be configured with pull-up or pull-down resistors,
- Operation from +5 volts only,
- BASIC, Turbo Pascal, and Turbo C source code; diagnostics program.

The following paragraphs briefly describe the major functions of the board. A more detailed discussion of board functions is included in Chapter 3, *Hardware Operation*, and Chapter 4, *Board Operation and Programming*. The board setup is described in Chapter 1, *Board Settings*.

Am9513A Timer/Counter

The versatile Am9513A general purpose timer/counter provides a variety of timing, sequencing, and counting functions. The Am9513A chip contains five 16-bit timer/counters which can be used individually or internally cascaded to form a counter of up to 80 bits. With 24 operating modes, up or down counting in binary or BCD, and hardware or software gating, these timer/counters can be easily tailored for a wide variety of applications. The timer/counters are clocked by an on-board 5 MHz crystal. The source, gate, and output for each timer/counter is available at the P2 I/O connector.

Digital I/O

The DM5804 has 24 TTL/CMOS-compatible digital I/O lines which can be directly interfaced with external devices or signals to sense switch closures, trigger digital events, or activate solid-state relays. These lines are provided by the on-board 8255 programmable peripheral interface chip. Pads for installing and activating pull-up or pull-down resistors are included on the board. Installation procedures are given at the end of Chapter 1, *Board Settings*.

AT Bus Connector J6 (DM6804)

The DM6804 is exactly the same as the DM5804 except for the addition of the AT bus connector J6. This allows you to stack the module with CPU's that have the AT bus connectors and access the AT interrupts.

What Comes With Your Board

You receive the following items in your DM5804 package:

- DM5804 interface board with stackthrough bus header
- Software and diagnostics diskette with BASIC, Turbo Pascal, and Turbo C source code
- User’s manual

If any item is missing or damaged, please call Real Time Devices’ Customer Service Department at (814) 234-8087. If you require service outside the U.S., contact your local distributor.

Board Accessories

In addition to the items included in your DM5804 package, RTD Embedded Technologies, Inc. offers a full line of board accessories. Call your local distributor or our main office for more information about these accessories and for help in choosing the best items to support your board’s application. Accessories for the DM5804 include the TB50 terminal board and XB50 prototype/terminal board for prototype development and easy signal access, the DM14 extender board for testing your module and XT50 twisted pair wire flat ribbon cable assembly for external interfacing.

Using This Manual

This manual is intended to help you install your new board and get it running quickly, while also providing enough detail about the board and its functions so that you can enjoy maximum use of its features even in the most complex applications. We assume that you already have an understanding of data acquisition and control principles and that you can customize the example software or write your own applications programs.

When You Need Help

This manual and the example programs in the software package included with your board provide enough information to properly use all of the board's features. If you have any problems installing or using this board, contact our Technical Support Department, (814) 234-8087, during regular business hours, eastern standard time or eastern daylight time, or send a FAX requesting assistance to (814) 234-5218. When sending a FAX request, please include your company's name and address, your name, your telephone number, and a brief description of the problem. You can also contact us through our E-mail address **techsupport@rtd.com**.

CHAPTER 1

BOARD SETTINGS

The DM5804 has jumper and switch settings you can change if necessary for your application. The board is factory-configured with the most often used settings. The factory settings are listed and shown on a diagram in the beginning of this chapter. Should you need to change these settings, use these easy-to-follow instructions before you install the board in your system.

Note that by installing resistor packs at four locations around the 8255 PPI and soldering jumpers in the desired locations on the associated pads, you can configure your 8255 digital I/O lines to be pulled up or pulled down. This procedure is explained at the end of this chapter.

Factory-Configured Switch and Jumper Settings

Table 1-1 lists the factory settings of the user-configurable jumper and switches on the DM5804. Figure 1-1 shows the board layout and the locations of the factory-set jumpers. The following paragraphs explain how to change the factory settings. Pay special attention to the setting of S1, the base address switch, to avoid address contention when you first use your board in your system.

Table 1-1 Factory Settings

Switch/ Jumper	Function Controlled	Factory Settings (Jumpers Installed)
P3	Connects one of the four sources selected on P4 to an interrupt channel; pulls tri-state buffer to ground (G) for multiple interrupt applications	G (ground for buffer); interrupt channels disabled
P4	Selects the interrupt source	EXTINT
S1	Sets the base address	300 hex (768 decimal)

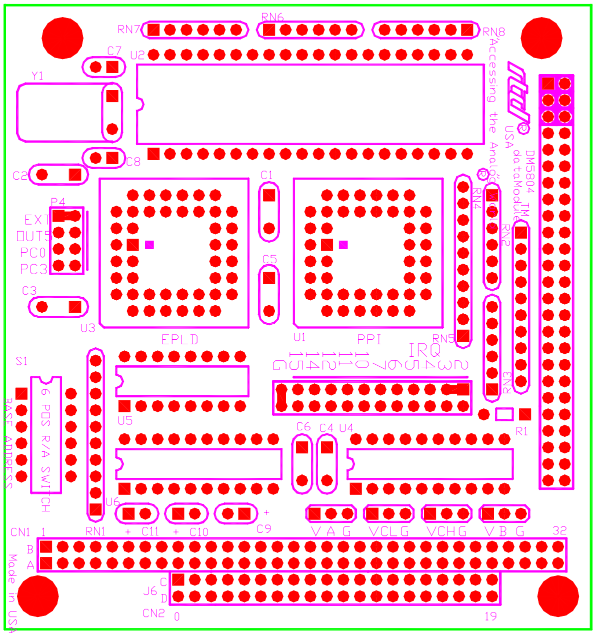


Fig. 1-1 — Board Layout Showing Factory-Configured Settings

P3 — Interrupt Channel Select (Factory Setting: G Connected, Interrupt Channels Disabled)

This header connector, shown in Figure 1-2, lets you connect an interrupt source selected on P4 to an interrupt channel, IRQ2 through IRQ7 . IRQ10, 11, 12, 14 and 15 can only be used if you have the DM6804 with the AT connector J6 installed. To connect the interrupt source selected on P4 to an IRQ channel, you must install a jumper across the desired IRQ channel. Figure 1-2a shows the factory setting and Figure 1-2b shows IRQ3 selected.

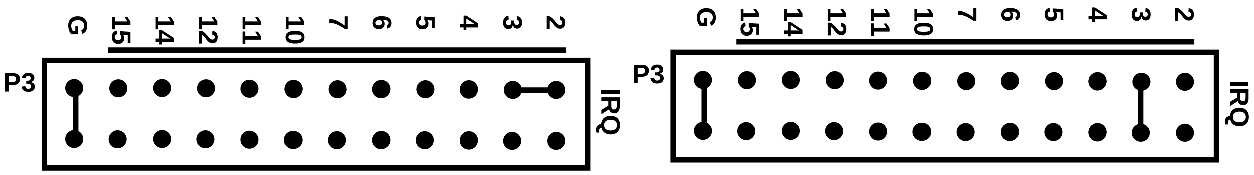


Fig. 1-2a — IRQ Disabled

Fig. 1-2b — IRQ3 Selected

Fig. 1-2 — Interrupt Channel Select Jumper, P3

The leftmost pair of pins on P3, labeled G, are provided so that you can install a jumper which connects a 1 kilohm pull-down resistor to the output of a high-impedance tri-state driver which carries the interrupt request signal. This pull-down resistor drives the interrupt request line low whenever interrupts are not active. So, whenever an interrupt request is made, the tri-state buffer is enabled, forcing the output high and causing an interrupt. You can monitor the interrupt status through bit 0 in the status word (I/O address location BA + 7). After the interrupt has been serviced, the clear command returns the IRQ line low, disabling the tri-state buffers, and pulling the output low again. Figure 1-3 shows this circuit. Because the interrupt request line is driven low only by the pull-down resistor, you can have two or more boards which share the same IRQ channel. You can tell which board issued the interrupt request by monitoring each board's IRQ status bit.

NOTE: When you use multiple boards that share the same interrupt, only one board should have the G jumper installed. The rest should be disconnected. Whenever you operate a single board, the G jumper should be installed.

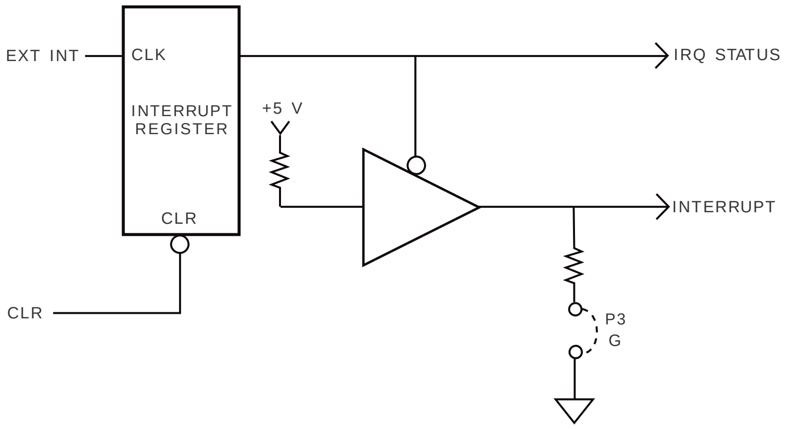


Fig. 1-3 — Pulling Down the Interrupt Request Line

P4 — Interrupt Source Select (Factory Setting: EXTINT)

This header connector, shown in Figure 1-4, lets you connect one of four interrupt sources for interrupt generation. These sources are: PC3, which is the INTRA signal from the 8255 PPI; PC0, which is the INTRB signal from the 8255 PPI; OUT5, the output from Am9513A timer/counter 5; and EXTINT, an external interrupt you can route onto the board through the P2 I/O connector. To connect an interrupt source, place the jumper across the desired set of pins. Note that only ONE interrupt source can be activated at a time.

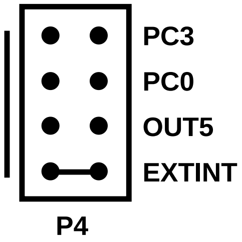


Fig. 1-4 — Interrupt Source Select Jumper, P4

S1 — Base Address (Factory Setting: 300 hex (768 decimal))

One of the most common causes of failure when you are first trying your board is address contention. Some of your computer’s I/O space is already occupied by internal I/O and other peripherals. When the DM5804 board attempts to use I/O address locations already used by another device, contention results and the board does not work.

To avoid this problem, the DM5804 has an easily accessible DIP switch, S1, which lets you select any one of 64 starting addresses in the computer’s I/O. Should the factory setting of 300 hex (768 decimal) be unsuitable for your system, you can select a different base address simply by setting the switches to any value shown in Table 1-2. The table shows the switch settings and their corresponding decimal and hexadecimal (in parentheses) values. Make sure that you verify the order of the switch numbers on the switch (1 through 6) before setting them. When the switches are pulled forward, they are OPEN, or set to logic 1, as labeled on the DIP switch package. When you set the base address for your board, record the value in the table inside the back cover. Figure 1-5 shows the DIP switch set for a base address of 300 hex (768 decimal).

Table 1-2 Base Address Switch Settings, S1

Base Address Decimal / (Hex)	Switch Setting 6 5 4 3 2 1	Base Address Decimal / (Hex)	Switch Setting 6 5 4 3 2 1	Base Address Decimal / (Hex)	Switch Setting 6 5 4 3 2 1	Base Address Decimal / (Hex)	Switch Setting 6 5 4 3 2 1
512 / (200)	0 0 0 0 0 0	640 / (280)	0 1 0 0 0 0	768 / (300)	1 0 0 0 0 0	896 / (380)	1 1 0 0 0 0
520 / (208)	0 0 0 0 0 1	648 / (288)	0 1 0 0 0 1	776 / (308)	1 0 0 0 0 1	904 / (388)	1 1 0 0 0 1
528 / (210)	0 0 0 0 1 0	656 / (290)	0 1 0 0 1 0	784 / (310)	1 0 0 0 1 0	912 / (390)	1 1 0 0 1 0
536 / (218)	0 0 0 0 1 1	664 / (298)	0 1 0 0 1 1	792 / (318)	1 0 0 0 1 1	920 / (398)	1 1 0 0 1 1
544 / (220)	0 0 0 1 0 0	672 / (2A0)	0 1 0 1 0 0	800 / (320)	1 0 0 1 0 0	928 / (3A0)	1 1 0 1 0 0
552 / (228)	0 0 0 1 0 1	680 / (2A8)	0 1 0 1 0 1	808 / (328)	1 0 0 1 0 1	936 / (3A8)	1 1 0 1 0 1
560 / (230)	0 0 0 1 1 0	688 / (2B0)	0 1 0 1 1 0	816 / (330)	1 0 0 1 1 0	944 / (3B0)	1 1 0 1 1 0
568 / (238)	0 0 0 1 1 1	696 / (2B8)	0 1 0 1 1 1	824 / (338)	1 0 0 1 1 1	952 / (3B8)	1 1 0 1 1 1
576 / (240)	0 0 1 0 0 0	704 / (2C0)	0 1 1 0 0 0	832 / (340)	1 0 1 0 0 0	960 / (3C0)	1 1 1 0 0 0
584 / (248)	0 0 1 0 0 1	712 / (2C8)	0 1 1 0 0 1	840 / (348)	1 0 1 0 0 1	968 / (3C8)	1 1 1 0 0 1
592 / (250)	0 0 1 0 1 0	720 / (2D0)	0 1 1 0 1 0	848 / (350)	1 0 1 0 1 0	976 / (3D0)	1 1 1 0 1 0
600 / (258)	0 0 1 0 1 1	728 / (2D8)	0 1 1 0 1 1	856 / (358)	1 0 1 0 1 1	984 / (3D8)	1 1 1 0 1 1
608 / (260)	0 0 1 1 0 0	736 / (2E0)	0 1 1 1 0 0	864 / (360)	1 0 1 1 0 0	992 / (3E0)	1 1 1 1 0 0
616 / (268)	0 0 1 1 0 1	744 / (2E8)	0 1 1 1 0 1	872 / (368)	1 0 1 1 0 1	1000 / (3E8)	1 1 1 1 0 1
624 / (270)	0 0 1 1 1 0	752 / (2F0)	0 1 1 1 1 0	880 / (370)	1 0 1 1 1 0	1008 / (3F0)	1 1 1 1 1 0
632 / (278)	0 0 1 1 1 1	760 / (2F8)	0 1 1 1 1 1	888 / (378)	1 0 1 1 1 1	1016 / (3F8)	1 1 1 1 1 1

0 = closed, 1 = open

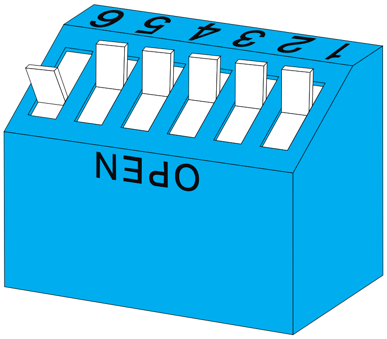


Fig. 1-5 — Base Address Switch, S1

Pull-up/Pull-down Resistors on Digital I/O Lines

The 8255 programmable peripheral interface provides 24 parallel TTL/CMOS compatible digital I/O lines which can be interfaced with external devices. The lines are divided into four groups: eight Port A lines, four Port C Lower lines, eight Port B lines, and four Port C Upper lines. You can install and connect pull-up or pull-down resistors for any or all of these four groups of lines. You may want to pull lines up for connection to switches. This will pull the line high when the switch is disconnected. Or, you may want to pull down lines connected to relays which control turning motors on and off. These motors turn on when the digital lines controlling them are high.

To use the pull-up/pull-down feature, you must first install 10 kilohm resistor packs in any or all of the four locations around the 8255, labeled PA, PB, PCL, and PCH. PA and PB take 10-pin packs, and CL and CH take 6-pin packs. Figure 1-6 shows a blowup of PA and PB.

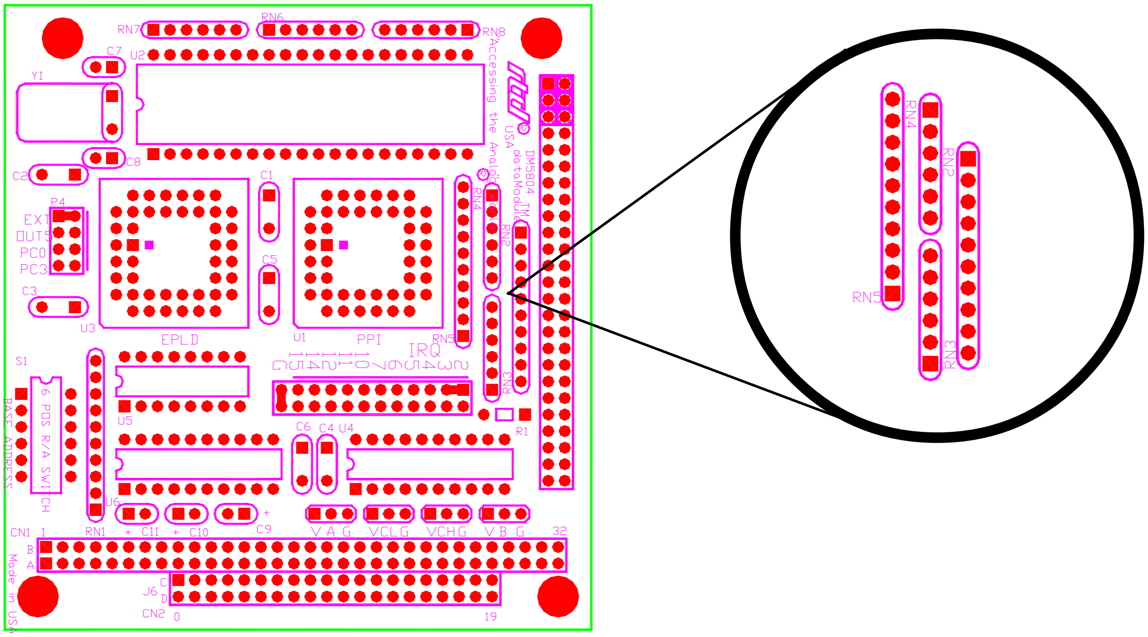


Fig. 1-6 — Port A and Port B Pull-up/Pull-down Resistor Circuitry

After the resistor packs are installed, you must connect them into the circuit as pull-ups or pull-downs. Locate the three-hole pads on the board near the resistor packs. They are labeled G (for ground) on one end and V (for Vcc) on the other end. The middle hole is common. PA is for Port A, PB for Port B, CL is for Port C Lower, and CH is for Port C Upper. Figure 1-6 shows a blowup of the pads for Port A and Port B. To operate as pull-ups, solder a jumper wire between the common pin (middle pin of the three) and the V pin. For pull-downs, solder a jumper wire between the common pin (middle pin) and the G pin. For example, Figure 1-7 shows Port A lines with pull-ups, Port C Lower with pull-downs, and Port C Upper with no resistors.

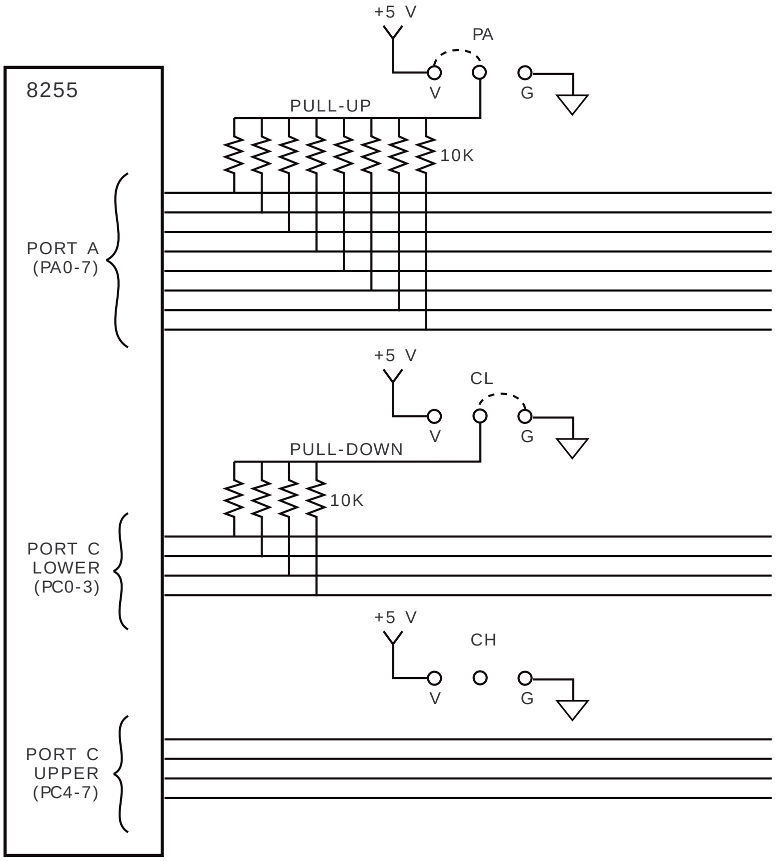


Fig. 1-7 — Adding Pull-ups and Pull-downs to Some Digital I/O Lines

CHAPTER 2

BOARD INSTALLATION

The DM5804 is easy to install in your cpuModule™ or other PC/104 based system. This chapter tells you step-by-step how to install and connect the board.

After you have installed the board and made all of your connections, you can turn your system on and run the 5804DIAG board diagnostics program included on your example software disk to verify that your board is working.

Board Installation

Keep the board in its antistatic bag until you are ready to install it in your cpuModule™ or other PC/104 based system. When removing it from the bag, hold the board at the edges and do not touch the components or connectors.

Before installing the board in your system, check the jumper and switch settings. Chapter 1 reviews the factory settings and how to change them. If you need to change any settings, refer to the appropriate instructions in Chapter 1. Note that incompatible jumper settings can result in unpredictable board operation and erratic response.

The DM5804 comes with a stackthrough P1 connector. The stackthrough connector lets you stack another board on top of your DM5804, plugging it into the data bus through the pins on the non-component side of the board.

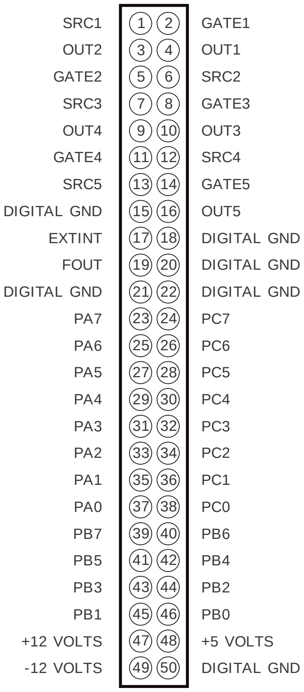
To install the board, follow the procedures described in the computer manual and the steps below:

1. Turn OFF the power to your system.
2. Touch the metal rack to discharge any static buildup and then remove the board from its antistatic bag.
3. Select the appropriate standoffs for your application to secure the board when you install it in your system (two sizes are included with the board).
4. Holding the board by its edges, orient it so that the P1 bus connector's pin 1 lines up with pin 1 of the expansion connector onto which you are installing the board.
5. After carefully positioning the board so that the DM5804's bus connector is resting on the expansion connector, gently and evenly press down on the board until it is secured on the connector.

   NOTE: Do not force the board onto the connector. If the board does not slide into place, remove it and try again. Wiggling the board or exerting too much pressure can result in damage to the DM5804 or to the module it is being stacked with.
6. After the board is installed, connect the cable to I/O connector P2 on the board. When making this connection, note that there is no keying to guide you in orientation. You must make sure that pin 1 of the cable is connected to pin 1 of P2 (pin 1 is marked on the board with a small square). For twisted pair cables, pin 1 is the dark brown wire; for standard single wire cables, pin 1 is the red wire.
7. Make sure all connections are secure.

External I/O Connections

Figure 2-1 shows the DM5804's P2 I/O connector pinout. Refer to this diagram as you make your I/O connections.



SRC1	1	2	GATE1
OUT2	3	4	OUT1
GATE2	5	6	SRC2
SRC3	7	8	GATE3
OUT4	9	10	OUT3
GATE4	11	12	SRC4
SRC5	13	14	GATE5
DIGITAL GND	15	16	OUT5
EXTINT	17	18	DIGITAL GND
FOUT	19	20	DIGITAL GND
DIGITAL GND	21	22	DIGITAL GND
PA7	23	24	PC7
PA6	25	26	PC6
PA5	27	28	PC5
PA4	29	30	PC4
PA3	31	32	PC3
PA2	33	34	PC2
PA1	35	36	PC1
PA0	37	38	PC0
PB7	39	40	PB6
PB5	41	42	PB4
PB3	43	44	PB2
PB1	45	46	PB0
+12 VOLTS	47	48	+5 VOLTS
-12 VOLTS	49	50	DIGITAL GND

Fig. 2-1 — P2 I/O Connector Pin Assignments

Connecting the Timer/Counters and Digital I/O

For all of the digital connections, the high side of an external signal source or destination device is connected to the appropriate signal pin on the I/O connector, and the low side is connected to any DIGITAL GND.

Running the 5804DIAG Diagnostics Program

Now that your board is ready to use, you will want to try it out. An easy-to-use, menu-driven diagnostics program, 5804DIAG, is included with your example software to help you verify your board's operation. You can also use this program to make sure that your current base address setting does not contend with another device.

CHAPTER 3

HARDWARE DESCRIPTION

This chapter describes the features of the DM5804 hardware. The major circuits are the timer/counters and the digital I/O lines. This chapter also describes the hardware-selectable interrupts.

The DM5804 has two major circuits, the timer/counters and the digital I/O lines. Figure 3-1 shows the block diagram of the board. This chapter describes the hardware which makes up the major circuits and hardware-selectable interrupts.

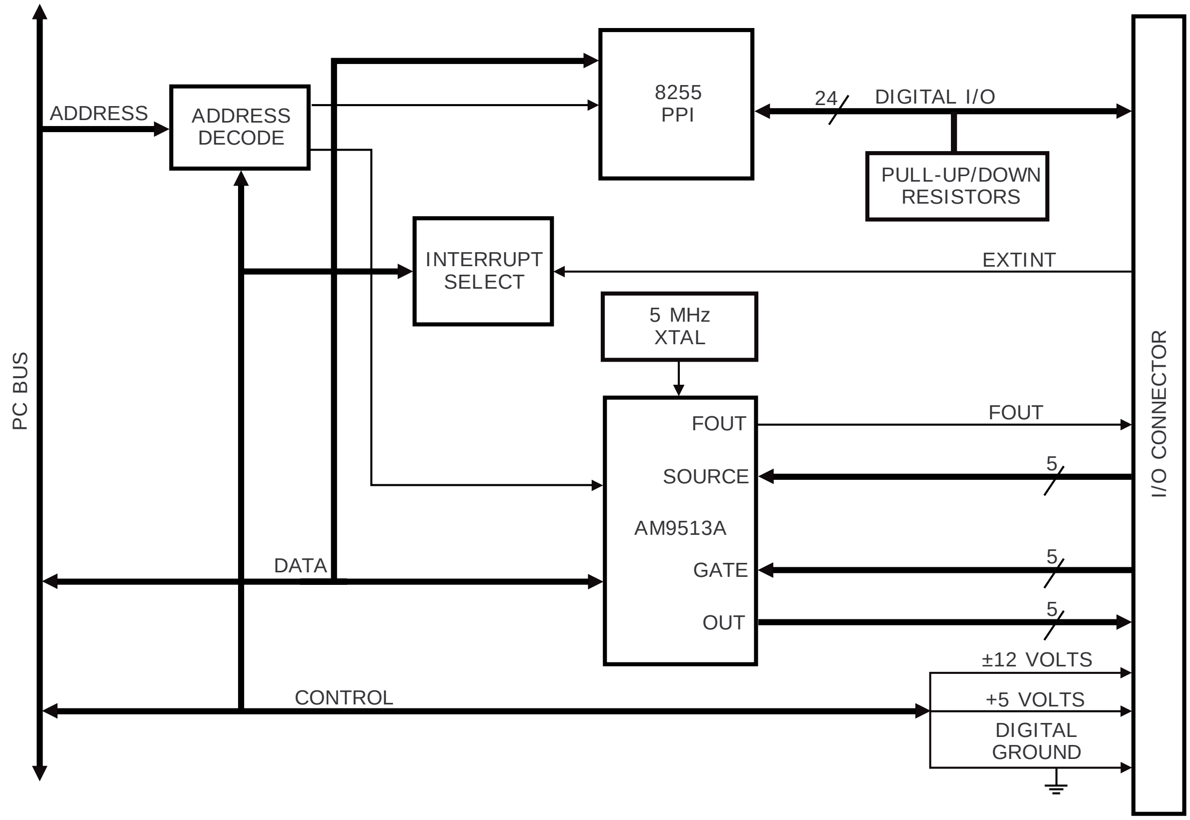


Fig. 3-1 DM804 Block Diagram

Am9513A Timer/Counters

The Am9513A System Timing Controller contains five general purpose 16-bit timer/counters which are capable of performing many different types of counting, sequencing, and timing functions. The Am9513A supports up or down counting in binary or BCD with hardware or software gating of each counter. Its 24 modes of operation are detailed in the Am9513A Data Sheet reprint from AMD included in Appendix C.

The Am9513A is structured with a series of internal registers that set the mode of operation for each counter. These registers are fully described in Appendix C.

Any of the counters can be internally cascaded to create a counter of up to 80 bits. For example, two cascaded counters form a 32-bit counter for longer counting capability. Rarely is it practical to cascade more than three counters. Cascading is described in Appendix C, Chapter 3 of the Am9513A data sheet.

The timer/counters are driven by an on-board 5 MHz crystal oscillator.

Digital I/O, Programmable Peripheral Interface

The programmable peripheral interface (PPI) is used for digital I/O functions. This high-performance TTL/CMOS compatible chip has 24 digital I/O lines divided into two groups of 12 lines each:

Group A — Port A (8 lines) and Port C Upper (4 lines);
Group B — Port B (8 lines) and Port C Lower (4 lines).

All three ports, A, B, and C, are available at the I/O connector, P2. You can use these ports in one of these three PPI operating modes:

Mode 0 — Basic input/output. Lets you use simple input and output operation for a port. Data is written to or read from the specified port.

Mode 1 — Strobed input/output. Lets you transfer I/O data from Port A or Port B in conjunction with strobes or handshaking signals.

Mode 2 — Strobed bidirectional input/output. Lets you communicate bidirectionally with an external device through Port A. Handshaking is similar to Mode 1.

These modes are detailed in the 8255 Data Sheet, reprinted from Intel in Appendix C.

Interrupts

The DM5804 has four jumper-selectable interrupt sources: PC3, which is the INTRA signal from the 8255 PPI; PC0, which is the INTRB signal from the 8255 PPI; OUT5, the output from Am9513A timer/counter 5; and EXTINT, an external interrupt you can route onto the board through I/O connector P2. Chapter 1 tells you how to set the jumpers on the interrupt header connectors, P3 and P4, and Chapter 4 describes how to program interrupts.

CHAPTER 4

BOARD OPERATION AND PROGRAMMING

This chapter shows you how to program your DM5804. It provides a complete description of the I/O map and a description of programming operations to aid you in programming. The example programs included on the disk in your board package are listed at the end of this chapter. These programs, written in Turbo C, Turbo Pascal, and BASIC, include source code to simplify your applications programming. Chapter 5 contains examples for setting up the timer/counters for specific applications.

Defining the I/O Map

The I/O map for the DM5804 is shown in Table 4-1 below. As shown, the board occupies eight consecutive I/O port locations. The base address (designated as BA) can be selected using DIP switch S1 as described in Chapter 1, *Board Settings*. This switch can be accessed without removing the board from the connector. The following sections describe the register contents of each address used in the I/O map.

Table 4-1: DM5804 I/O Map

Register Description	Read Function	Write Function	Address * (Decimal)
8255 PPI Port A	Read Port A digital input lines	Program Port A digital output lines	BA + 0
8255 PPI Port B	Read Port B digital input lines	Program Port B digital output lines	BA + 1
8255 PPI Port C	Read Port C digital input lines	Program Port C digital output lines	BA + 2
8255 PPI Control Word	Reserved	Program PPI configuration	BA + 3
Am9513A Data Word	Read data register	Program data register	BA + 4
Am9513A Control Word	Read control register	Program control register	BA + 5
IRQ Enable	Reserved	Enable and disable interrupt generation	BA + 6
Interrupt Status/Clear	Read status of interrupt	Clear interrupt	BA + 7
* BA = Base Address			

BA + 0: PPI Port A — Digital I/O (Read/Write)

Transfers the 8-bit Port A digital input and digital output data between the board and an external device. A read transfers data from the external device, through P2, and into PPI Port A; a write transfers the written data from Port A through P2 to an external device.

BA + 1: PPI Port B — Digital I/O (Read/Write)

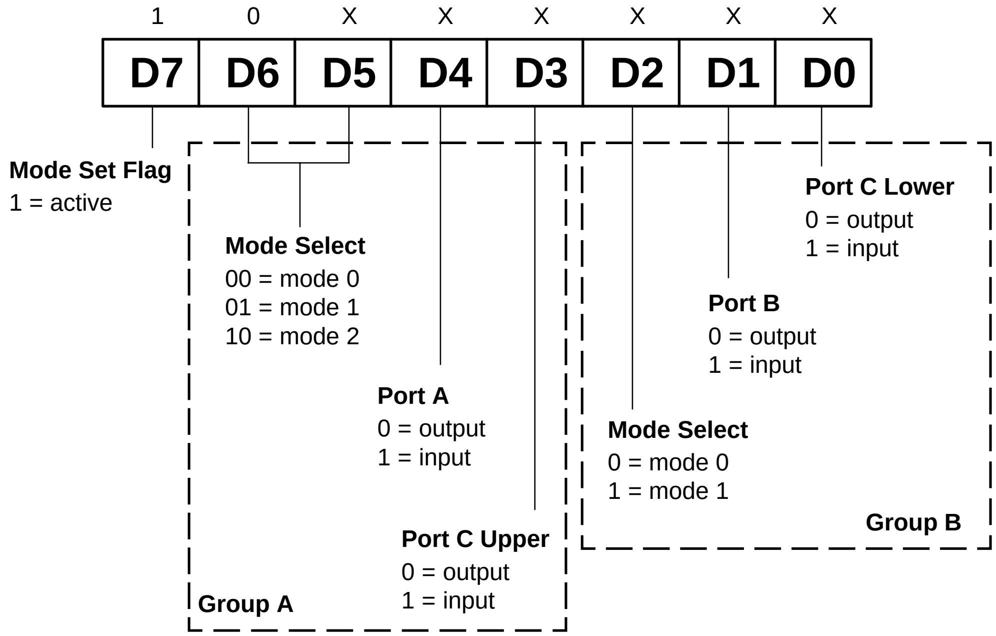
Transfers the 8-bit Port B digital input and digital output data between the board and an external device. A read transfers data from the external device, through P2, and into PPI Port B; a write transfers the written data from Port B through P2 to an external device.

BA + 2: PPI Port C — Digital I/O (Read/Write)

Transfers the two 4-bit Port C digital input and digital output data groups (Port C Upper and Port C Lower) between the board and an external device. A read transfers data from the external device, through P2, and into PPI Port C; a write transfers the written data from Port C through P2 to an external device.

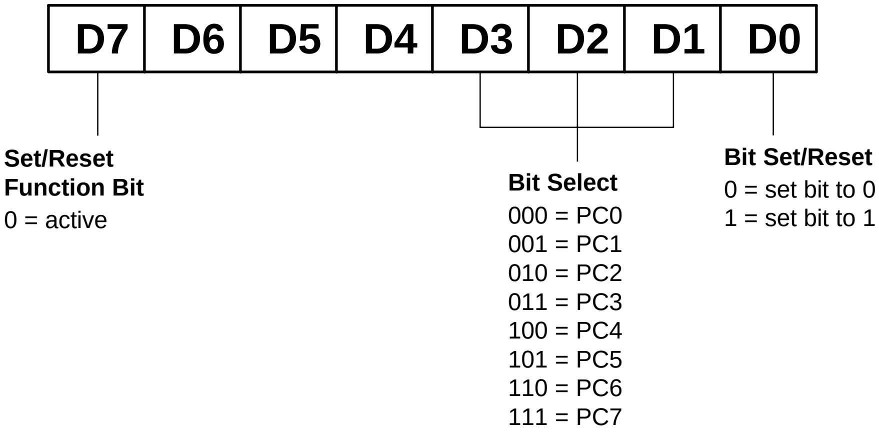
BA + 3: 8255 PPI Control Word (Write Only)

When bit 7 of this word is set to 1, a write programs the PPI configuration. The table below shows the control words for the 16 possible Mode 0 Port I/O combinations.

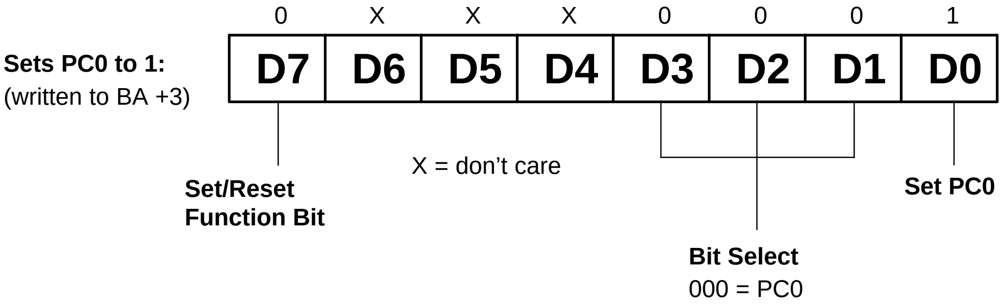


8255 Port I/O Flow Direction and Control Words, Mode 0						
Group A		Group B		Control Word		
Port A	Port C Upper	Port B	Port C Lower	Binary	Decimal	Hex
Output	Output	Output	Output	1 0 0 0 0 0 0 0	128	80
Output	Output	Output	Input	1 0 0 0 0 0 0 1	129	81
Output	Output	Input	Output	1 0 0 0 0 0 1 0	130	82
Output	Output	Input	Input	1 0 0 0 0 0 1 1	131	83
Output	Input	Output	Output	1 0 0 0 1 0 0 0	136	88
Output	Input	Output	Input	1 0 0 0 1 0 0 1	137	89
Output	Input	Input	Output	1 0 0 0 1 0 1 0	138	8A
Output	Input	Input	Input	1 0 0 0 1 0 1 1	139	8B
Input	Output	Output	Output	1 0 0 1 0 0 0 0	144	90
Input	Output	Output	Input	1 0 0 1 0 0 0 1	145	91
Input	Output	Input	Output	1 0 0 1 0 0 1 0	146	92
Input	Output	Input	Input	1 0 0 1 0 0 1 1	147	93
Input	Input	Output	Output	1 0 0 1 1 0 0 0	152	98
Input	Input	Output	Input	1 0 0 1 1 0 0 1	153	99
Input	Input	Input	Output	1 0 0 1 1 0 1 0	154	9A
Input	Input	Input	Input	1 0 0 1 1 0 1 1	155	9B

When bit 7 of this word is set to 0, a write can be used to individually program the Port C lines.



For example, if you want to set Port C bit 0 to 1, you would set up the control word so that bit 7 is 0; bits 1, 2, and 3 are 0 (this selects PC0); and bit 0 is 1 (this sets PC0 to 1). The control word is set up like this:



BA + 4: Am9513A Data Register (Read/Write)

Accesses the Am9513A data register. See data sheet included in Appendix C for more information on the operation of the Am9513A.

IMPORTANT

Because of the bus release time of the Am9513A, AMD recommends you insert a small delay between software accesses to the chip.

BA + 5: Am9513A Command Register (Read/Write)

Accesses the Am9513A command register. See data sheet included in Appendix C for more information on the operation of the Am9513A.

IMPORTANT

Because of the bus release time of the Am9513A, AMD recommends you insert a small delay between software accesses to the chip.

BA + 6: IRQ Enable (Write Only)

Enables and disables interrupt generation. Writing a “1” enables interrupt generation; writing a “0” disables interrupt generation.

0	0	0	0	0	0	0	
D7	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**

Interrupt Enable/Disable
0 = interrupt disabled
1 = interrupt enabled

BA + 7: Interrupt Status/Clear (Read/Write)

A read shows the status of the interrupt (bit 0 only) as defined below. A write clears the interrupt (data written is irrelevant). Each time the interrupt status bit goes high, a write should follow to clear the bit.

X	X	X	X	X	X	X	
D7	**D6**	**D5**	**D4**	**D3**	**D2**	**D1**	**D0**

Interrupt Status
0 = no interrupt
1 = interrupt has occurred

Programming the DM5804

This section gives you some general information about programming and the DM5804 board. Chapter 5 provides some specific programming examples, and the Am9513A data sheet in Appendix C provides detailed programming information for all 24 operating modes of the Am9513A. These tools will help you as you use the example programs included with the board. All of the program descriptions in this section use decimal values unless otherwise specified.

The DM5804 is programmed by writing to and reading from the correct I/O port locations on the board. These I/O ports were defined in the previous section. Most high-level languages such as BASIC, Pascal, C, and C++, and of course assembly language, make it very easy to read/write these ports. The table below shows you how to read from and write to I/O ports using some popular programming languages.

Language	Read	Write
BASIC	Data=INP(Address)	OUT Address,Data
Turbo C	Data=inportb(Address)	outportb(Address,Data)
Turbo Pascal	Data:=Port[Address]	Port[Address]:=Data
Assembly	mov dx,Address in al,dx	mov dx,Address mov al,Data out dx,al

In addition to being able to read/write the I/O ports on the DM5804, you must be able to perform a variety of operations that you might not normally use in your programming. The table below shows you some of the operators discussed in this section, with an example of how each is used with Pascal, C, and BASIC. Note that the modulus operator is used to retrieve the least significant byte (LSB) of a two-byte word, and the integer division operator is used to retrieve the most significant byte (MSB).

Language	Modulus	Integer Division	AND	OR
C	% a = b % c	/ a = b / c	& a = b & c	| a = b | c
Pascal	MOD a := b MOD c	DIV a := b DIV c	AND a := b AND c	OR a := b OR c
BASIC	MOD a = b MOD c	\ a = b \ c	AND a = b AND c	OR a = b OR c

Many compilers have functions that can read/write either 8 or 16 bits from/to an I/O port. For example, Turbo Pascal uses **Port** for 8-bit port operations and **PortW** for 16 bits, Turbo C uses **inportb** for an 8-bit read of a port and **inport** for a 16-bit read. **Be sure to use only 8-bit operations with the DM5804!**

Clearing and Setting Bits in a Port

When you clear or set one or more bits in a port, you must be careful that you do not change the status of the other bits. You can preserve the status of all bits you do not wish to change by proper use of the AND and OR binary operators. Using AND and OR, single or multiple bits can be easily cleared in one operation.

To **clear** a single bit in a port, AND the current value of the port with the value b, where b = 255 - 2^{bit}.

> **Example:** Clear bit 5 in a port. Read in the current value of the port, AND it with 223 (223 = 255 - 2^5), and then write the resulting value to the port. In BASIC, this is programmed as:

```
V = INP(PortAddress)
V = V AND 223
OUT PortAddress, V
```

To **set** a single bit in a port, OR the current value of the port with the value b, where b = 2^{bit}.

> **Example:** Set bit 3 in a port. Read in the current value of the port, OR it with 8 (8 = 2^3), and then write the resulting value to the port. In Pascal, this is programmed as:

```
V := Port[PortAddress];
V := V OR 8;
Port[PortAddress] := V;
```

Setting or clearing more than one bit at a time is accomplished just as easily. To **clear** multiple bits in a port, AND the current value of the port with the value b, where b = 255 - (the sum of the values of the bits to be cleared). Note that the bits do not have to be consecutive.

> **Example:** Clear bits 2, 4, and 6 in a port. Read in the current value of the port, AND it with 171 (171 = 255 - 2^2 - 2^4 - 2^6), and then write the resulting value to the port. In C, this is programmed as:

```
v = inportb(port_address);
v = v & 171;
outportb(port_address, v);
```

To **set** multiple bits in a port, OR the current value of the port with the value b, where b = the sum of the individual bits to be set. Note that the bits to be set do not have to be consecutive.

> **Example:** Set bits 3, 5, and 7 in a port. Read in the current value of the port, OR it with 168 (168 = 2^3 + 2^5 + 2^7), and then write the resulting value back to the port. In assembly language, this is programmed as:

```
mov dx, PortAddress
in al, dx
or al, 168
out dx, al
```

Often, assigning a range of bits is a mixture of setting and clearing operations. You can set or clear each bit individually or use a faster method of first clearing all the bits in the range then setting only those bits that must be set using the method shown above for setting multiple bits in a port. The following example shows how this two-step operation is done.

> **Example:** Assign bits 3, 4, and 5 in a port to 101 (bits 3 and 5 set, bit 4 cleared). First, read in the port and clear bits 3, 4, and 5 by ANDing them with 199. Then set bits 3 and 5 by ORing them with 40, and finally write the resulting value back to the port. In C, this is programmed as:

```
v = inportb(port_address);
v = v & 199;
v = v | 40;
outportb(port_address, v);
```

A final note: Don't be intimidated by the binary operators AND and OR and try to use operators for which you have a better intuition. For instance, if you are tempted to use addition and subtraction to set and clear bits in place of the methods shown above, DON'T! Addition and subtraction may seem logical, but they **will not work** if you try to clear a bit that is already clear or set a bit that is already set. For example, you might think that to set bit 5 of a port, you simply need to read in the port, add 32 (2^5) to that value, and then write the resulting value back to the port. This works fine if bit 5 is not already set. But, what happens when bit 5 *is* already set? Bits 0 to 4 will be unaffected and we can't say for sure what happens to bits 6 and 7, but we can say for sure that bit 5 ends up cleared instead of being set. A similar problem happens when you use subtraction to clear a bit in place of the method shown above.

Now that you know how to clear and set bits, we are ready to look at the programming steps for the DM5804 board functions.

Initializing the Am9513A

The Am9513A has a sophisticated internal architecture which is programmed through a series of internal registers. These internal registers are accessed by writing to and reading from only two I/O port locations: the Data Register port at BA + 4 and the Control Register port at BA + 5. In our example programs, we follow these steps to initialize the Am9513A:

1. Send a master reset to the Am9513A
2. Point to and set up the master mode register
3. Point to and set up counter 1 mode register
4. Point to counter 1 load register and load desired value
5. Point to and set up counter 2 mode register
6. Point to counter 2 load register and load desired value
   -
   -

11. Point to and set up counter 5 mode register
12. Point to counter 5 load register and load desired value
13. Load and arm counters

The examples on the disk and in Chapter 5 will aid you in programming the Am9513A for your application. These tools and the data sheet in Appendix C provide a comprehensive description of timer/counter operation.

IMPORTANT

> Because of the bus release time of the Am9513A, AMD recommends you insert a small delay between software accesses to the chip.

Initializing the 8255

Before you can use the digital I/O lines on your DM5804, the 8255 PPI must be initialized. This step must be executed every time you start up, reset, or reboot your computer.

The 8255 is initialized by writing the appropriate control word to I/O port BA + 3. The contents of your control word will vary, depending on how you want to configure your I/O lines. Use the control word description in the previous I/O map section to help you program the right value. In the example below, a decimal value of 128 sets up the 8255 so that all I/O lines are Mode 0 outputs.

1	0	0	0	0	0	0	0
D7	D6	D5	D4	D3	D2	D1	D0

Digital I/O Operations

Once the 8255 is initialized, you can use the digital I/O line to control or monitor external devices.

Interrupts

- What Is an Interrupt?

An interrupt is an event that causes the processor in your computer to temporarily halt its current process and execute another routine. Upon completion of the new routine, control is returned to the original routine at the point where its execution was interrupted.

Interrupts are very handy for dealing with asynchronous events (events that occur at less than regular intervals). Keyboard activity is a good example; your computer cannot predict when you might press a key and it would be a waste of processor time for it to do nothing while waiting for a keystroke to occur. Thus, the interrupt scheme is used and the processor proceeds with other tasks. Then, when a keystroke does occur, the keyboard ‘interrupts’ the processor, and the processor gets the keyboard data, places it in memory, and then returns to what it was doing before it was interrupted. Other common devices that use interrupts are modems, disk drives, and mice.

Your DM5804 board can interrupt the processor when one of the four interrupt sources is enabled through the jumper settings on P3 and P4. By using this interrupt, you can write software that effectively deals with real world events.

- Interrupt Request Lines

To allow different peripheral devices to generate interrupts on the same computer, the PC bus has eight different interrupt request (IRQ) lines. A transition from low to high on one of these lines generates an interrupt request which is handled by the PC’s interrupt controller. The interrupt controller checks to see if interrupts are to be acknowledged from that IRQ and, if another interrupt is already in progress, it decides if the new request should supersede the one in progress or if it has to wait until the one in progress is done. This prioritizing allows an interrupt to be interrupted if the second request has a higher priority. The priority level is based on the number of the IRQ; IRQ0 has the highest priority, IRQ1 is second-highest, and so on through IRQ7, which has the lowest. Many of the IRQs are used by the standard system resources. IRQ0 is used by the system timer, IRQ1 is used by the keyboard, IRQ3 by COM2, IRQ4 by COM1, and IRQ6 by the disk drives. Therefore, it is important for you to know which IRQ lines are available in your system for use by the DM5804 board.

- 8259 Programmable Interrupt Controller

The chip responsible for handling interrupt requests in the PC is the 8259 Programmable Interrupt Controller. To use interrupts, you will need to know how to read and set the 8259’s interrupt mask register (IMR) and how to send the end-of-interrupt (EOI) command to the 8259.

- Interrupt Mask Register (IMR)

Each bit in the interrupt mask register (IMR) contains the mask status of an IRQ line; bit 0 is for IRQ0, bit 1 is for IRQ1, and so on. If a bit is **set** (equal to 1), then the corresponding IRQ is masked and it will not generate an interrupt. If a bit is **clear** (equal to 0), then the corresponding IRQ is unmasked and can generate interrupts. The IMR is programmed through port 21H.

IRQ7	**IRQ6**	**IRQ5**	**IRQ4**	**IRQ3**	**IRQ2**	**IRQ1**	**IRQ0**

I/O Port 21H

For all bits:
0 = IRQ unmasked (enabled)
1 = IRQ masked (disabled)

- End-of-Interrupt (EOI) Command

After an interrupt service routine is complete, the 8259 interrupt controller must be notified. This is done by writing the value 20H to I/O port 20H.

- What Exactly Happens When an Interrupt Occurs?

Understanding the sequence of events when an interrupt is triggered is necessary to properly write software interrupt handlers. When an interrupt request line is driven high by a peripheral device (such as the DM5804), the interrupt controller checks to see if interrupts are enabled for that IRQ, and then checks to see if other interrupts are active or requested and determines which interrupt has priority. The interrupt controller then interrupts the processor. The current code segment (CS), instruction pointer (IP), and flags are pushed on the stack for storage, and a new CS and IP are loaded from a table that exists in the lowest 1024 bytes of memory. This table is referred to as the interrupt vector table and each entry is called an interrupt vector. Once the new CS and IP are loaded from the interrupt vector table, the processor begins executing the code located at CS:IP. When the interrupt routine is completed, the CS, IP, and flags that were pushed on the stack when the interrupt occurred are now popped from the stack and execution resumes from the point where it was interrupted.

- Using Interrupts in Your Programs

Adding interrupts to your software is not as difficult as it may seem, and what they add in terms of performance is often worth the effort. Note, however, that although it is not that hard to use interrupts, the smallest mistake will often lead to a system hang that requires a reboot. This can be both frustrating and time-consuming. But, after a few tries, you'll get the bugs worked out and enjoy the benefits of properly executed interrupts. In addition to reading the following paragraphs, study the source code included on your DM5804 program disk in the interrupt programs for a better understanding of interrupt program development.

- Writing an Interrupt Service Routine (ISR)

The first step in adding interrupts to your software is to write the interrupt service routine (ISR). This is the routine that will automatically be executed each time an interrupt request occurs on the specified IRQ. An ISR is different than standard routines that you write. First, on entrance, the processor registers should be pushed onto the stack **BEFORE** you do anything else. Second, just before exiting your ISR, you must clear the interrupt status of the DM5804 and write an end-of-interrupt command to the 8259 controller. Finally, when exiting the ISR, in addition to popping all the registers you pushed on entrance, you must use the IRET instruction and **not** a plain RET. The IRET automatically pops the flags, CS, and IP that were pushed when the interrupt was called.

If you find yourself intimidated by interrupt programming, take heart. Most Pascal and C compilers allow you to identify a procedure (function) as an interrupt type and will automatically add these instructions to your ISR, with one important exception: most compilers **do not** automatically add the end-of-interrupt command to the procedure; you must do this yourself. Other than this and the few exceptions discussed below, you can write your ISR just like any other routine. It can call other functions and procedures in your program and it can access global data. If you are writing your first ISR, we recommend that you stick to the basics; just something that will convince you that it works, such as incrementing a global variable.

NOTE: If you are writing an ISR using assembly language, you are responsible for pushing and popping registers and using IRET instead of RET.

There are a few cautions you must consider when writing your ISR. The most important is, **do not use any DOS functions or routines that call DOS functions from within an ISR**. DOS is **not** reentrant; that is, a DOS function cannot call itself. In typical programming, this will not happen because of the way DOS is written. But what about when using interrupts? Then, you could have a situation such as this in your program. If DOS function X is being executed when an interrupt occurs and the interrupt routine makes a call to DOS function X, then function X is essentially being called while it is already active. Such a reentrancy attempt spells disaster because DOS functions are not written to support it. This is a complex concept and you do not need to understand it. Just make sure that you do not call any DOS functions from within your ISR. The one wrinkle is that, unfortunately, it is not obvious which library routines included with your compiler use DOS functions. A rule of thumb is that routines which write to the screen, or check the status of or read the keyboard, and any disk I/O routines use DOS and should be avoided in your ISR.

The same problem of reentrancy exists for many floating point emulators as well, meaning you may have to avoid floating point (real) math in your ISR.

Note that the problem of reentrancy exists, no matter what programming language you are using. Even if you are writing your ISR in assembly language, DOS and many floating point emulators are not reentrant. Of course, there are ways around this problem, such as those which involve checking to see if any DOS functions are currently active when your ISR is called, but such solutions are well beyond the scope of this discussion.

The second major concern when writing your ISR is to make it as short as possible in terms of execution time. Spending long periods of time in your ISR may mean that other important interrupts are being ignored. Also, if you spend too long in your ISR, it may be called again before you have completed handling the first run. This often leads to a hang that requires a reboot.

Your ISR should have this structure:

- Push any processor registers used in your ISR. Most C and Pascal interrupt routines automatically do this for you.
- Put the body of your routine here.
- Clear the interrupt bit on the DM5804 by writing any value to BA + 7
- Issue the EOI command to the 8259 interrupt controller by writing 20H to port 20H.
- Pop all registers pushed on entrance. Most C and Pascal interrupt routines automatically do this for you.

The following C and Pascal examples show what the shell of your ISR should be like:

In C:

```
void interrupt ISR(void)
{
   /* Your code goes here. Do not use any DOS functions! */
   outportb(BaseAddress + 7, 0);       /* Clear DM5804 interrupt */
   outportb(0x20, 0x20);               /* Send EOI command to 8259 */
}
```

In Pascal:

```
Procedure ISR; Interrupt;
begin
   { Your code goes here. Do not use any DOS functions! }
   Port[BaseAddress + 7] := 0;      { Clear DM5804 interrupt }
   Port[$20] := $20;                { Send EOI command to 8259 }
end;
```

- Saving the Startup Interrupt Mask Register (IMR) and Interrupt Vector

The next step after writing the ISR is to save the startup state of the interrupt mask register and the interrupt vector that you will be using. The IMR is located at I/O port 21H. The interrupt vector you will be using is located in the interrupt vector table which is simply an array of 256-bit (4-byte) pointers and is located in the first 1024 bytes of memory (Segment = 0, Offset = 0). You can read this value directly, but it is a better practice to use DOS function 35H (get interrupt vector). Most C and Pascal compilers provide a library routine for reading the value of a vector. The vectors for the hardware interrupts are vectors 8 through 15, where IRQ0 uses vector 8, IRQ1 uses vector 9, and so on. Thus, if the DM5804 will be using IRQ3, you should save the value of interrupt vector 11.

Before you install your ISR, temporarily mask out the IRQ you will be using. This prevents the IRQ from requesting an interrupt while you are installing and initializing your ISR. To mask the IRQ, read in the current IMR at I/O port 21H and **set** the bit that corresponds to your IRQ (remember, setting a bit disables interrupts on that IRQ while clearing a bit enables them). The IMR is arranged so that bit 0 is for IRQ0, bit 1 is for IRQ1, and so on. See the paragraph entitled *Interrupt Mask Register (IMR)* earlier in this chapter for help in determining your IRQ's bit. After setting the bit, write the new value to I/O port 21H.

With the startup IMR saved and the interrupts on your IRQ temporarily disabled, you can assign the interrupt vector to point to your ISR. Again, you can overwrite the appropriate entry in the vector table with a direct memory write, but this is a bad practice. Instead, use either DOS function 25H (set interrupt vector) or, if your compiler provides it, the library routine for setting an interrupt vector. Remember that vector 8 is for IRQ0, vector 9 is for IRQ1, and so on.

If you need to program the source of your interrupts, do that next. For example, if you are using the programmable interval timer to generate interrupts, you must program it to run in the proper mode and at the proper rate.

Finally, clear the bit in the IMR for the IRQ you are using. This enables interrupts on the IRQ.

– Restoring the Startup IMR and Interrupt Vector

Before exiting your program, you must restore the interrupt mask register and interrupt vectors to the state they were in when your program started. To restore the IMR, write the value that was saved when your program started to I/O port 21H. Restore the interrupt vector that was saved at startup with either DOS function 35H (get interrupt vector), or use the library routine supplied with your compiler. Performing these two steps will guarantee that the interrupt status of your computer is the same after running your program as it was before your program started running.

- Common Interrupt Mistakes

- Remember that hardware interrupts are numbered 8 through 15, even though the corresponding IRQs are numbered 0 through 7.
- Two of the most common mistakes when writing an ISR are forgetting to clear the interrupt status of the DM5804 and forgetting to issue the EOI command to the 8259 interrupt controller before exiting the ISR.

Example Programs

Included with the DM5804 is a set of example programs that demonstrate the use of many of the board's features. These examples are in written in C, Pascal, and BASIC. Also included is an easy-to-use menu-driven diagnostics program, 804DIAG, which is especially helpful when you are first checking out your board after installation.

C and Pascal Programs

These programs are source code files so that you can easily develop your own custom software for your DM5804.

Timer/Counter:

INTRPTS Shows how to generate interrupts and read the digital I/O lines.

COUNT Shows how to use the Am9513A as a simple counter.

Digital I/O:

DIGITAL Simple program that shows how to read and write the digital I/O lines.

BASIC Programs

These programs are source code files so that you can easily develop your own custom software for your DM5804.

Timer/Counter:

COUNT Shows how to use the Am9513A as a simple counter.

FCOUNT Shows how to create a frequency counter using the Am9513A.

Digital I/O:

DIGITAL Simple program that shows how to read and write the digital I/O lines.

CHAPTER 5

EXAMPLES OF Am9513A APPLICATIONS

This chapter steps through some example programs to help you understand how the Am9513A registers are programmed. The data pointer register and command registers are summarized in tables. The master mode and counter mode register bit assignments are also included, as well as the frequency scaler ratios.

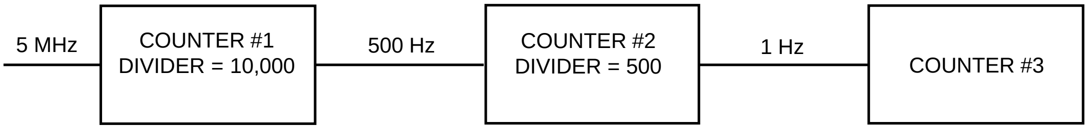
This chapter provides a more detailed look at some example programs using the Am9513A for counting and timing functions. If you are unfamiliar with the Am9513A and how it is programmed, walking through these examples and the other example programs included on your DM5804 disk may be the best way to understand the many registers and their operation so that you can successfully develop your own programs for your specific applications.

IMPORTANT

> Because of the bus release time of the Am9513A, AMD recommends you insert a small delay between software accesses to the chip.

EXAMPLE: Counting Program Using Timer/Counters 1, 2, and 3

This BASIC program, COUNT on the example disk, shows you how to program the Am9513A's timer/ counters 1, 2, and 3 to perform a simple counting function. In this example, counter 1 is used to divide the on-board 5 MHz clock by 10,000. The output from counter 1 (5 MHz ÷ 10,000 = 500 Hz) is used to clock counter 2. Counter 2 is used to divide this 500 Hz clock by 500. The result is a 1 Hz clock which is used to clock counter 3. Counter 3 counts the 1 Hz pulses. The count value from counter 3 is displayed on the screen. This value should start at 0 and increment once each second.



The first lines of the program clear the screen and set up the base address of the DM5804. The address in the variable "BA" must match the setting of the base address switch, S1, on the board. The factory setting of S1 is 300 hex (768 decimal).

```
CLS
INPUT "ENTER BASE ADDRESS IN DECIMAL: "; BA
```

The next section of the program sets up the computer screen:

```
CLS                           'CLEAR SCREEN
DIM RESULT AS LONG            'DIMENSION VARIABLE "RESULT" AS A LONG INTEGER
KEY(1) ON                     'TURN F1 KEY ON
LOCATE 2, 25
PRINT "DM5804 COUNTER DEMO PROGRAM !";
LOCATE 10, 31
PRINT "COUNTER #3 VALUE:";
LOCATE 24, 2
PRINT "F1 = QUIT ";
```

The next section of the program sets up the address for the DM5804 registers. These addresses are defined at the beginning of Chapter 4.

```
PA = BA + 0                   'ADDRESS FOR 8255 PORT A
PB = BA + 1                   'ADDRESS FOR 8255 PORT B
PC = BA + 2                   'ADDRESS FOR 8255 PORT C
CW = BA + 3                   'ADDRESS FOR 8255 CONTROL WORD
DR = BA + 4                   'ADDRESS FOR AM9513A COUNTER DATA REGISTER
CR = BA + 5                   'ADDRESS FOR AM9513A COUNTER CONTROL REGISTER
IRQEN = BA + 6                'ADDRESS FOR INTERRUPT ENABLE
STAT = BA + 7                 'ADDRESS TO READ INTERRUPT STATUS
IRQCLR = BA + 7               'ADDRESS TO CLEAR INTERRUPT STATUS BIT
```

Now, reset the Am9513A timer/counter chip (see Table 5-2):

```
OUT CR, &HFF                  'AM9513A MASTER RESET
```

Table 5-1 Load Data Pointer Commands

	Element Cycle			Hold Cycle
	Mode Register	Load Register	Hold Register	Hold Register
Counter 1	01	09	11	19
Counter 2	02	0A	12	1A
Counter 3	03	0B	13	1B
Counter 4	04	0C	14	1C
Counter 5	05	0D	15	1D
Master Mode Register = 17 Alarm 1 Register = 07 Alarm 2 Register = 0F Status Register = 1F				
NOTE: All codes are in hex.				

Next, set up the Am9513A master mode register (see Figure 5-1). These are the settings we will use:

Scaler Control = binary division
Data Pointer Control = disable increment
Data Bus Width = 8 bits
FOUT Gate = FOUT on
FOUT Divider = divide by 16
FOUT Source = F1 (see Figure 5-3)
Compare 2 Enable = disabled
Compare 1 Enable = disabled
Time-of-Day Mode = disabled

VALUE = HEX 4000

```
OUT CR, &H17                  ‘POINT TO MASTER MODE REGISTER (TABLE 5-1)
OUT DR, &H0                   ‘MASTER MODE LSB
OUT DR, &H40                  ‘MASTER MODE MSB
```

Next, set up the counter 1 mode register (see Figure 5-2). These are the settings we will use:

Gating Control = no gating
Source Edge = rising edge
Count Source Selection = F1
Count Control = disable special gate
= reload from load
= count repetitively
= binary count
= count down
Output Control = TC toggled

VALUE = HEX 0B22

```
OUT CR, &H1                   ‘POINT TO COUNTER 1 MODE REGISTER (TABLE 5-1)
OUT DR, &H22                  ‘COUNTER 1 MODE LSB
OUT DR, &H0B                  ‘COUNTER 1 MODE MSB
```

Table 5-2 Am9513A Command Summary

Command Code								
C7	**C6**	**C5**	**C4**	**C3**	**C2**	**C1**	**C0**	**Command Description**
0	0	0	E2	E1	G4	G2	G1	Load data pointer register with contents of E & G fields. (E - 000, G - 110). E & G fields described in Appendix C.
0	0	1	S5	S4	S3	S2	S1	Arm counting for all selected counters
0	1	0	S5	S4	S3	S2	S1	Load contents of specified source into all selected counters
0	1	1	S5	S4	S3	S2	S1	Load & arm all selected counters*
1	0	0	S5	S4	S3	S2	S1	Disarm & save all selected counters
1	0	1	S5	S4	S3	S2	S1	Save all selected counters in hold register
1	1	0	S5	S4	S3	S2	S1	Disarm all selected counters
1	1	1	0	1	N4	N2	N1	Set toggle out (high) for counter N (001 ² N ² 101)
1	1	1	0	0	N4	N2	N1	Clear toggle out (low) for counter N (001 ² N ² 101)
1	1	1	1	0	N4	N2	N1	Step counter N (001 ² N ² 101)
1	1	1	0	1	0	0	0	Set MM14 (disable data pointer sequencing)
1	1	1	0	1	1	1	0	Set MM12 (gate off FOUT)
1	1	1	0	1	1	1	1	Set MM13 (enter 16-bit bus mode)
1	1	1	0	0	0	0	0	Clear MM14 (enable data pointer sequencing)
1	1	1	0	0	1	1	0	Clear MM12 (gate on FOUT)
1	1	1	0	0	1	1	1	Clear MM13 (enter 8-bit bus mode)
1	1	1	1	1	0	0	0	Enable prefetch for write operations
1	1	1	1	1	0	0	1	Disable prefetch for write operations
1	1	1	1	1	1	1	1	Master reset

* Not to be used for asynchronous operations.

Put the hex number 2710 (decimal 10,000) in counter 1 load register:

```
OUT CR, &H9                    'POINT TO COUNTER 1 LOAD REGISTER (TABLE 5-1)
OUT DR, &H10                   'COUNTER 1 LSB
OUT DR, &H27                   'COUNTER 1 MSB
```

Next, set up the counter 2 mode register (see Figure 5-2). These are the settings we will use:

Gating Control = no gating
Source Edge = rising edge
Count Source Selection = TCN-1
Count Control = disable special gate
= reload from load
= count repetitively
= binary count
= count down
Output Control = TC toggled

VALUE = HEX 0022

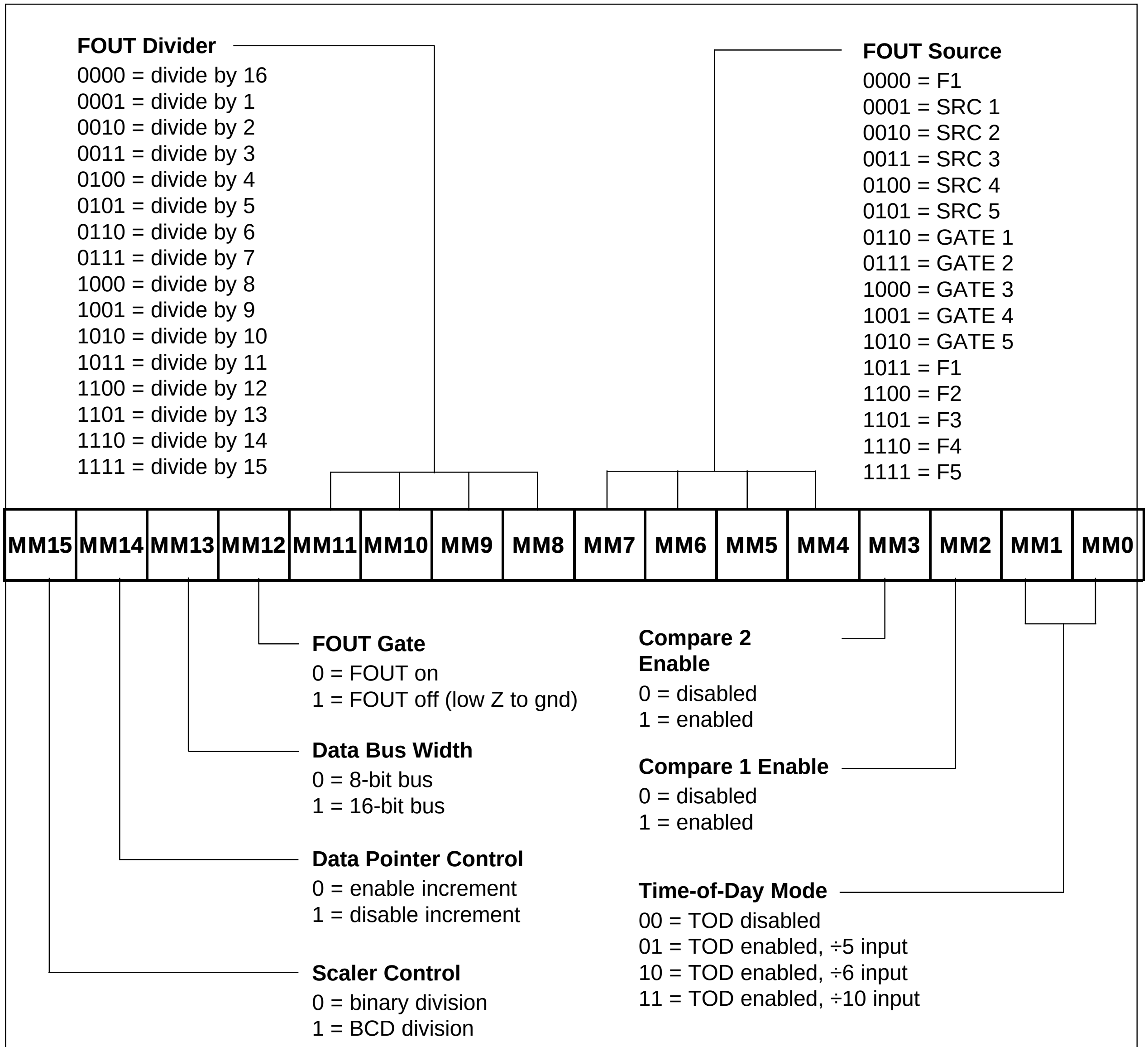


Fig. 5-1 — Master Mode Register Bit Assignments

```
OUT CR, &H2                    'POINT TO COUNTER 2 MODE REGISTER (TABLE 5-1)
OUT DR, &H22                   'COUNTER 2 MODE LSB
OUT DR, &H0                    'COUNTER 2 MODE MSB
```

Put the hex number 1F4 (decimal 500) in counter 2 load register:

```
OUT CR, &HA                    'POINT TO COUNTER 2 LOAD REGISTER (TABLE 5-1)
OUT DR, &HF4                   'COUNTER 2 LSB
OUT DR, &H1                    'COUNTER 2 MSB
```

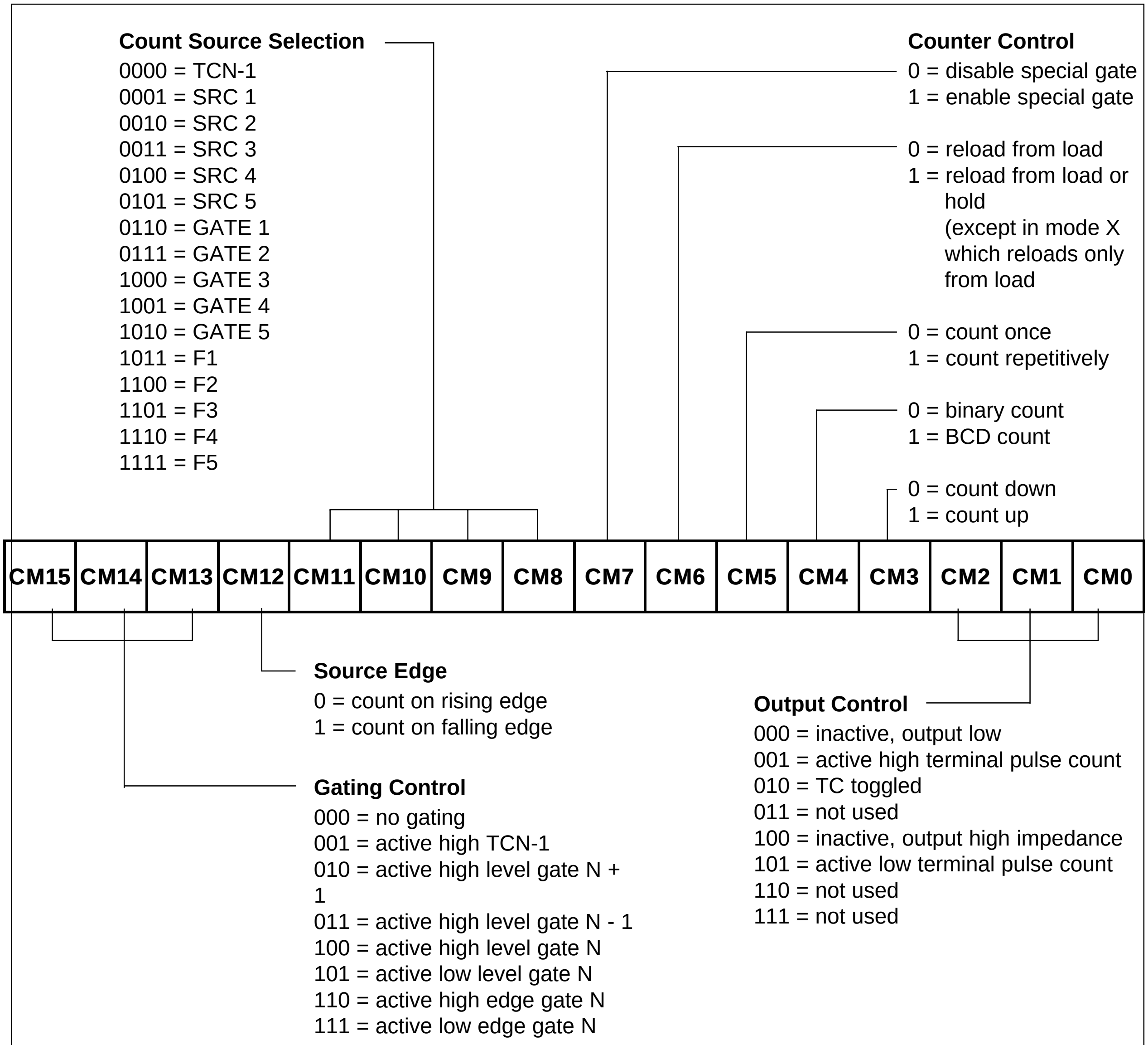


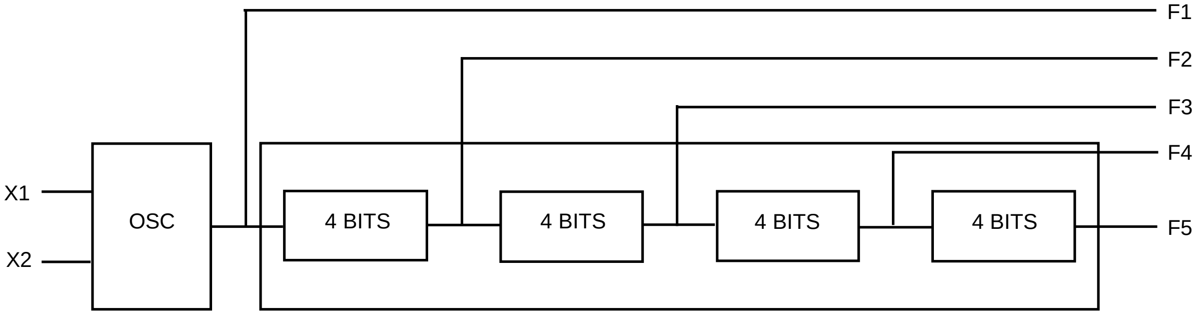
Fig. 5-2 — Counter Mode Register Bit Assignments

Next, set up the counter 3 mode register (see Figure 5-2). These are the settings we will use:

Gating Control = no gating
Source Edge = rising edge
Count Source Selection = TCN-1
Count Control = disable special gate
= reload from load
= count repetitively
= binary count
= count up
Output Control = TC toggled

VALUE = HEX 002A

```
OUT CR, &H3                 ‘POINT TO COUNTER 3 MODE REGISTER (TABLE 5-1)
OUT DR, &H2A                ‘COUNTER 3 MODE LSB
OUT DR, &H0                 ‘COUNTER 3 MODE MSB
```



Frequency	BCD Scaling (MM15 = 1)		Binary Scaling (MM15 = 0)	
	Ratio	With On-board 5 MHz Clock	Ratio	With On-board 5 MHz Clock
F1	OSC	5 MHz	OSC	5 MHz
F2	F1 / 10	500 kHz	F1 / 16	312.5 kHz
F3	F1 / 100	50 kHz	F1 / 256	19.53 kHz
F4	F1 / 1000	5 kHz	F1 / 4096	1.221 kHz
F5	F1 / 10,000	500 Hz	F1 / 65,536	76.3 Hz

Fig. 5-3 — Frequency Scaler Ratio

Put the hex number 0000 in counter 3 load register:

```
OUT CR, &HB                  ‘POINT TO COUNTER 3 LOAD REGISTER (TABLE 5-1)
OUT DR, &H0                  ‘COUNTER 3 LSB
OUT DR, &H0                  ‘COUNTER 3 MSB

OUT CR, &H67                 ‘LOAD & ARM COUNTERS 1,2 & 3 (TABLE 5-2)
```

The main program is:

```
OUT CR, &HA4                 ‘SAVE COUNTER 3 IN HOLD REGISTER (TABLE 5-2)
OUT CR, &H13                 ‘POINT TO COUNTER 3 HOLD REGISTER (TABLE 5-1)
LSB = INP(DR)                ‘READ COUNTER 3 LSB
MSB = INP(DR)                ‘READ COUNTER 3 MSB
RESULT = LSB + (MSB * 256)   ‘COMBINE LSB & MSB
LOCATE 12, 37
PRINT USING “#####”; RESULT
ON KEY(1) GOSUB QUIT
GOTO MAIN
```

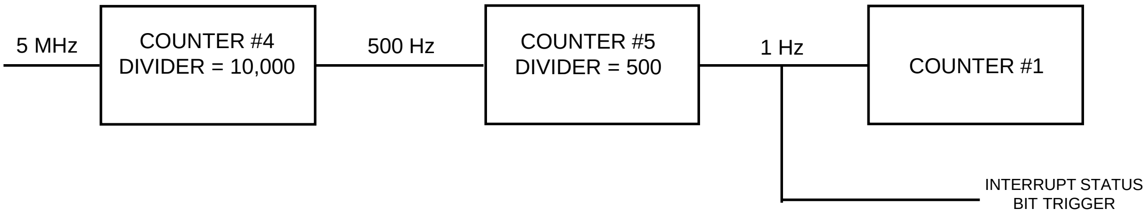
To quit:

```
KEY(1) OFF
END
```

EXAMPLE: Setting Up the Am9513A as a Frequency Counter

This program, FCOUNT on the example disk, shows you how to program the Am9513A as a simple frequency counter. In this example, counter 4 is used to divide the on-board 5 MHz clock by 10,000. The output from counter 4 (5 MHz ÷ 10,000 = 500 Hz) is used to clock counter 5. Counter 5 is used to divide this 500 Hz clock by 500. The result is a 1 Hz output which is used to gate counter 1. This 1 Hz output is also used to trigger the interrupt status bit. The program monitors this status bit to determine when to read counters 1 and 2. Counters 1 and 2 are used to count the SRC1 input for 1 second intervals.

NOTE: For this program to operate properly, a jumper must be connected between OUT5 (pin 16) and GATE1 (pin 2) at I/O connector P2.



The first lines of the program clear the screen and set up the base address of the DM5804. The address in the variable “BA” must match the setting of the base address switch, S1, on the board. The factory setting of S1 is 300 hex (768 decimal).

```
CLS
INPUT “ENTER BASE ADDRESS IN DECIMAL: “; BA
```

The next section of the program sets up the computer screen:

```
CLS                           ‘CLEAR SCREEN
DIM RESULT AS DOUBLE          ‘DIMENSION VARIABLE “RESULT”
KEY(1) ON                     ‘TURN F1 KEY ON
LOCATE 2, 25
PRINT “DM5804 FREQUENCY COUNTER DEMO PROGRAM !”;
LOCATE 24, 2
PRINT “F1 = QUIT “;
```

The next section of the program sets up the address for the DM5804 registers. These addresses are defined at the beginning of Chapter 4.

```
PA = BA + 0                   ‘ADDRESS FOR 8255 PORT A
PB = BA + 1                   ‘ADDRESS FOR 8255 PORT B
PC = BA + 2                   ‘ADDRESS FOR 8255 PORT C
CW = BA + 3                   ‘ADDRESS FOR 8255 CONTROL WORD
DR = BA + 4                   ‘ADDRESS FOR AM9513A COUNTER DATA REGISTER
CR = BA + 5                   ‘ADDRESS FOR AM9513A COUNTER CONTROL REGISTER
IRQEN = BA + 6                ‘ADDRESS FOR INTERRUPT ENABLE
STAT = BA + 7                 ‘ADDRESS TO READ INTERRUPT STATUS
IRQCLR = BA + 7               ‘ADDRESS TO CLEAR INTERRUPT STATUS BIT
```

Next, you must enable status generation:

```
OUT IRQEN, 1
```

Now, reset the Am9513A timer/counter chip (see Table 5-2):

```
OUT CR, &HFF                  ‘AM9513A MASTER RESET
```

Next, set up the Am9513A master mode register (see Figure 5-1). These are the settings we will use:

Scaler Control = binary division
Data Pointer Control = enable increment
Data Bus Width = 8 bits
FOUT Gate = FOUT on
FOUT Divider = divide by 16
FOUT Source = F1 (see Figure 5-3)
Compare 2 Enable = disabled
Compare 1 Enable = disabled
Time-of-Day Mode = disabled

VALUE = HEX 0000

```
OUT CR, &H17                  ‘POINT TO MASTER MODE REGISTER (TABLE 5-1)
OUT DR, &H0                   ‘MASTER MODE LSB
OUT DR, &H0                   ‘MASTER MODE MSB
```

Next, set up the counter 1 mode register (see Figure 5-2). These are the settings we will use:

Gating Control = active low gate n
Source Edge = rising edge
Count Source Selection = SRC1
Count Control = disable special gate
= reload from load
= count repetitively
= binary count
= count up
Output Control = active high TC

VALUE = HEX A129

```
OUT CR, &H1                   ‘POINT TO COUNTER 1 MODE REGISTER (TABLE 5-1)
OUT DR, &H29                  ‘COUNTER 1 MODE LSB
OUT DR, &HA1                  ‘COUNTER 1 MODE MSB
```

Put the hex number 0000 (decimal 0) in counter 1 load register:

```
OUT CR, &H9                   ‘POINT TO COUNTER 1 LOAD REGISTER (TABLE 5-1)
OUT DR, &H0                   ‘COUNTER 1 LSB
OUT DR, &H0                   ‘COUNTER 1 MSB
```

Next, set up the counter 2 mode register (see Figure 5-2). These are the settings we will use:

Gating Control = no gating
Source Edge = rising edge
Count Source Selection = TCN-1
Count Control = disable special gate
= reload from load
= count repetitively
= binary count
= count up
Output Control = active high TC

VALUE = HEX 0029

```
OUT CR, &H2                   ‘POINT TO COUNTER 2 MODE REGISTER (TABLE 5-1)
OUT DR, &H29                  ‘COUNTER 2 MODE LSB
OUT DR, &H0                   ‘COUNTER 2 MODE MSB
```

Put the hex number 0000 (decimal 0) in counter 2 load register:

```
OUT CR, &HA                    ‘POINT TO COUNTER 2 LOAD REGISTER (TABLE 5-1)
OUT DR, &H0                    ‘COUNTER 2 LSB
OUT DR, &H0                    ‘COUNTER 2 MSB
```

Next, set up the counter 4 mode register (see Figure 5-2). These are the settings we will use:

Gating Control = no gating
Source Edge = rising edge
Count Source Selection = F1
Count Control = disable special gate
= reload from load
= count repetitively
= binary count
= count down
Output Control = TC toggled

VALUE = HEX 0B22

```
OUT CR, &H4                    ‘POINT TO COUNTER 4 MODE REGISTER (TABLE 5-1)
OUT DR, &H22                   ‘COUNTER 4 MODE LSB
OUT DR, &H0B                   ‘COUNTER 4 MODE MSB
```

Put the hex number 2710 (decimal 10,000) in counter 4 load register:

```
OUT CR, &HC                    ‘POINT TO COUNTER 4 LOAD REGISTER (TABLE 5-1)
OUT DR, &H10                   ‘COUNTER 4 LSB
OUT DR, &H27                   ‘COUNTER 4 MSB
```

Next, set up the counter 5 mode register (see Figure 5-2). These are the settings we will use:

Gating Control = no gating
Source Edge = rising edge
Count Source Selection = TCN-1
Count Control = disable special gate
= reload from load
= count repetitively
= binary count
= count down
Output Control = TC toggled

VALUE = HEX 0022

```
OUT CR, &H5                    ‘POINT TO COUNTER 5 MODE REGISTER (TABLE 5-1)
OUT DR, &H22                   ‘COUNTER 5 MODE LSB
OUT DR, &H0                    ‘COUNTER 5 MODE MSB
```

Put the hex number 1F4 (decimal 500) in counter 5 load register:

```
OUT CR, &HD                    ‘POINT TO COUNTER 5 LOAD REGISTER (TABLE 5-1)
OUT DR, &HF4                   ‘COUNTER 5 LSB
OUT DR, &H1                    ‘COUNTER 5 MSB

OUT IRQCLR, 0                  ‘CLEAR INTERRUPT STATUS
OUT CR, &H7B                   ‘LOAD & ARM COUNTERS 1,2,4 & 5 (TABLE 5-2)
```

The main program is:

```
IRQ = INP(STAT) AND 1
IF IRQ <> 1 GOTO MAIN          ‘CHECK IF INTERRUPT STATUS = 1
OUT IRQCLR, 0                  ‘CLEAR INTERRUPT STATUS
OUT CR, &H83                   ‘DISARM & SAVE COUNTERS 1 & 2 (TABLE 5-2)
OUT CR, &H63                   ‘LOAD & ARM COUNTERS 1 & 2 (TABLE 5-2)
OUT CR, &H11                   ‘POINT TO COUNTER 1 HOLD REGISTER (TABLE 5-1)
LSB(1) = INP(DR)               ‘READ COUNTER 1 LSB
MSB(1) = INP(DR)               ‘READ COUNTER 1 MSB
LSB(2) = INP(DR)               ‘READ COUNTER 2 LSB
MSB(2) = INP(DR)               ‘READ COUNTER 2 MSB
RESULT = LSB(1) + (MSB(1) * 256) + (LSB(2) * 256 ^ 2) + (MSB(2) * 256 ^ 3)
LOCATE 12, 35
PRINT USING “##########”; RESULT;
PRINT "Hz"
ON KEY(1) GOSUB QUIT
GOTO MAIN
```

To quit:

```
KEY(1) OFF
END
```

APPENDIX A

DM5804/DM6804 SPECIFICATIONS

DM5804/DM6804 Characteristics Typical @ 25° C

Interface

cpuModule™ & other PC/104 form factor compatible
Switch-selectable base address, I/O mapped
Jumper-selectable interrupts

Digital I/O ... CMOS 82C55 (Optional NMOS 8255)

Number of lines	24
Logic compatibility	TTL/CMOS (Configurable with optional I/O pull-up/pull-down resistors)
High-level output voltage	4.2V, min
Low-level output voltage	0.45V, max
High-level input voltage	2.2V, min; 5.5V, max
Low-level input voltage	-0.3V, min; 0.8V, max
Input load current	±10 µA
Input capacitance, C(IN)@F=1MHz	10 pF
Output capacitance, C(OUT)<@F=1MHz	20 pF

Timer/Counter ... Am9513A

Five 16-bit timer/counters
Binary or BCD up or down counting

Programmable operating modes	24

Counter input source	External clock (6.9 MHz, max);
	on-board 5 MHz clock;
	external gate input; or
	adjacent counter output

Counter outputs	Available externally; used as PC interrupts or internally cascaded to adjacent counter
Counter gate source	External input; counter output; or software control

Miscellaneous Inputs/Outputs

+5 volts/digital ground (PC bus-sourced)
External interrupt input
Frequency output
Additional gate inputs

Current Requirements

+5 volts	220 mA, max

P2 Connector

50-pin right angle header

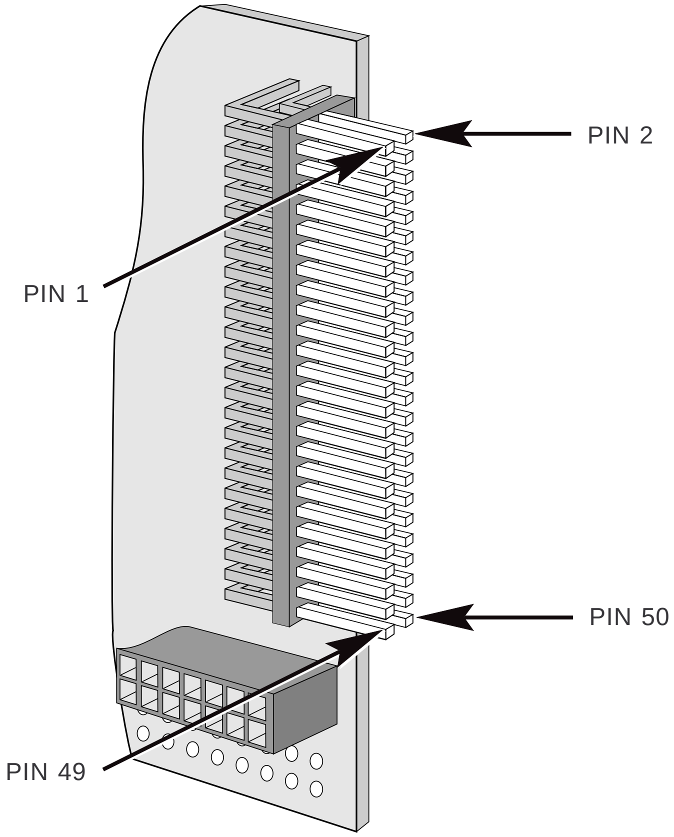
Size

3.55"L x 3.775"W x 0.6" H (90mm x 96mm x 16mm)

APPENDIX B

P2 CONNECTOR PIN ASSIGNMENTS

SRC1	1	2	GATE1
OUT2	3	4	OUT1
GATE2	5	6	SRC2
SRC3	7	8	GATE3
OUT4	9	10	OUT3
GATE4	11	12	SRC4
SRC5	13	14	GATE5
DIGITAL GND	15	16	OUT5
EXTINT	17	18	DIGITAL GND
FOUT	19	20	DIGITAL GND
DIGITAL GND	21	22	DIGITAL GND
PA7	23	24	PC7
PA6	25	26	PC6
PA5	27	28	PC5
PA4	29	30	PC4
PA3	31	32	PC3
PA2	33	34	PC2
PA1	35	36	PC1
PA0	37	38	PC0
PB7	39	40	PB6
PB5	41	42	PB4
PB3	43	44	PB2
PB1	45	46	PB0
+12 VOLTS	47	48	+5 VOLTS
-12 VOLTS	49	50	DIGITAL GND



P2 Mating Connector Part Numbers	
Manufacturer	**Part Number**
AMP	1-746094-0
3M	3425-7650

APPENDIX C

COMPONENT DATA SHEETS

AMD Am9513A System Timing Controller
Data Sheet Reprint

Celeritous Technical Services Corp
3308 34th St
Lubbock, Texas 79410

800.687.6510 / 806.793.0708
FAX 806.793.0710
http://www.celeritous.com

CTS9513-2
5 Chan 16 bit 20MHz Counter/Timer

FUNCTIONS

- Five 16 bit programmable up/down counters
- Programmable Pulse Generation
- Programmable Delay Generator
- Pulse Measurement
- Event Counting
- Frequency Measurement
- System Synchronization
- Real Time Clock

APPLICATIONS

- Computer System Timing
  - Real Time Clock with Alarm
  - Watchdog Timer
  - Programmable System/Bus Clock
  - Wait State Generation
- Data Acquisition
  - Programmable Converter Clock
  - Pulse Measurement
  - Frequency Counter
  - Event Counter
- ATE
  - Programmable Stimulus Generator
  - Timing Extremes Generator
- Laser Systems
  - Timing Sequencer
  - Programmable Delay Generator
  - External Equipment Synchronization
  - Burst Mode Generator
- Industrial Process Control
  - Pulse Frequency Sensor conversion
  - System Timing/Synchronization

EXTENDED FEATURES

- Up to 20 MHz Maximum input frequency
- Lower Power

STANDARD AM9513 FEATURES

- Five independent 16 bit counters
- Up/Down, Binary/BCD Counting
- Internal Binary/BCD Prescaling
- One Shot/Continuous Outputs
- Software/External triggering
- Tri-state Outputs
- Programmable output polarities
- Programmable gate polarities/edges
- Time of Day/Alarm Functions
- Programmable Internal/External Counter Source
- Fully AM9513 Hardware/Software Compatible
- Dual count registers on each counter

Figure 1 - CTS9513 DIP-40 Package

CTS9513 OVERVIEW

For two decades the most flexible counter/timer peripheral device available was the Advanced Micro Devices AM9513 Counter Timer. Until discontinued in 1995 the AM9513 was a leading device in industrial and scientific timing controllers. Its only limitation was its 7 Mhz maximum clock speed…………..*until now*...........

Building on over two decades of successful use as the most flexible programmable counter/timer device, the CTS9513 breaks the old limitations of the AM9513 in a new technology device with over 3 times the speed of the venerable '9513 with 16 bit counters. Sporting up to a 20 MHz maximum Input clock, the CTS9513 allows timing resolutions of 50 ns and gate pulses as short as 50nS. This opens up a whole new range of capabilities and applications for this device.

The CTS9513 is an ideal solution for direct replacement or new designs. With its CMOS construction it consumes far less power and runs much cooler than the original NMOS device. Due to its ASIC construction it can not be obsoleted

The CTS9513 is Hardware and Software compatible with the AM9513, allowing use of your present software drivers. Standard Packaging for the CTS9513 is the DIP-40, PLCC-44

OTHER PRODUCTS

Celeritous Technical Service specializes in the creation of replacements for discontinued and obsolete ICs. Using the latest in ASIC technology and EDA Design Tools, Celeritous Technical can provide rapid, high quality, cost effective form, fit and function replacements for obsolete digital ICs. Visit us on the web at http://www.celeritous.com for more information on our products and services.

DEVICE DESCRIPTION

The CTS9513 is a custom, high speed ASIC implementation of the AMD AM9513 System Timing Controller. The '9513 has long been the most versatile counter/timer peripheral device, featuring far more flexibility than competing timing devices such as the Intel 8253/8254, Motorola 6840 or others. A large installed base of devices and software drivers already exists.

The principal limitation of the AM9513 was its maximum frequency limitation of 7 Mhz imposed by its late 1970's NMOS LSI design. The CTS9513 shatters this barrier with a 20 MHz maximum clock speed and much lower power consumption due to its CMOS construction.

The CTS9513 Counter/Timer is capable of a wide variety of applications including, but not limited to:

- Event Counting
- Event Sequencing
- Programmable pulse generation
- Programmable delay generation
- Frequency counting
- Frequency synthesis
- Real Time Clock
- Alarm Clock Functions
- Watchdog Timing
- Retriggerable Pulse Generation
- Non-Retriggerable Pulse Generation
- Waveform Analysis
- Interrupt Generation
- Pulse burst generation

The user has control over key features such as:

- Output Polarities
- Output Impedance
- Input Trigger, Edge Polarities
- Hardware gating/triggering
- Software gating/triggering
- Count Up/Down
- BCD/Binary Counting
- Real time count register read
- Internal counter concatenation (up to 80 bits)
- Programmable frequency source selection
- Programmable internal clock pre-scaling

FEATURES

BACKWARDS COMPATIBLE

The CTS9513 maintains backwards compatibility with most AM9513 features, allowing continued use of your existing software drivers. Data may be transferred in 8 or 16 bit increments. All internal data paths in the CTS9513 are 16 bit. All '9513 commands registers and modes are supported.

PACKAGING

Figure 2 illustrates the DIP-40 Package pinout of the device which conforms to the original AM9513 pinouts.

Table 2 summarizes the pinouts of the PLCC-44 package illustrated in Figure 3 which conform to the original AM9513 PLCC pinouts.

SIGNALS

The following signal names and description conform to the original AM9513 device.

VCC
+5 Volt Power Supply

VSS
Ground

X1
The CTS9513 does not provide an internal crystal oscillator and must be driven from an external source. X1 should be left open

X2
X2 should be connected to an external TTL source and pulled up to VCC

FOUT (Frequency Divider Outputs)
The FOUT line is generated by internally programmable counters. The clock source for these counters may be any of the external GATE or SOURCE inputs as well as any of the internally prescaled clock outputs.

SOURCE1-5 (Count Source Inputs)
Source inputs 1-5 provide external clock source lines which may be routed to any of the internal counters or the FOUT divider. The active count edge for the source is programmed at the counter.

Symbol	Description	Min	Max	Units
V_{DD}	DC Supply Voltage	-0.3	7	Volts
V_{IN}	Input Voltage at Any Pin	-0.3	V_{DD}+.3	Volts
T_{OP}	Operating Temperature AxI	-40	85	^{o} C
T_{ST}	Storage Temperature	-55	150	^{o} C

Table 1. Absolute Maximum Ratings

PLCC-44 Package Pinouts			
Pin	**Signal**	**Pin**	**Signal**
1	VCC	23	D8
2	OUT2	24	VSS
3	NC	25	D9
4	OUT1	26	D10
5	GATE1	27	D11
6	X1	28	D12
7	X2	29	D13
8	FOUT	30	D14
9	NC	31	D15
10	C/D	32	NC
11	WR	33	SOURCE5
12	CS	34	SOURCE4
13	RD	35	SOURCE3
14	NC	36	SOURCE2
15	D0	37	SOURCE1
16	D1	38	GATE5
17	D2	39	GATE4
18	D3	40	GATE3
19	D4	41	OUT5
20	D5	42	OUT4
21	D6	43	GATE2
22	D7	44	OUT3

Table 2. PLCC-44 Pinouts

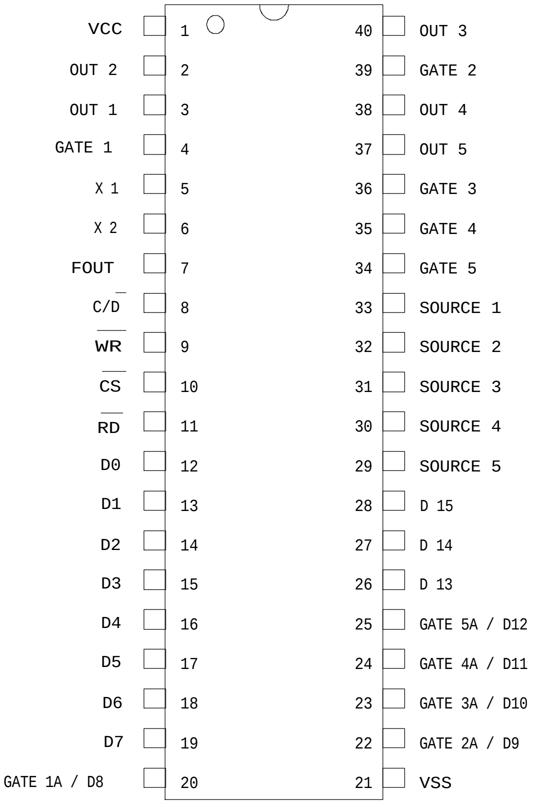


Figure 2 - CTS9513 DIP-40 Package Pinouts

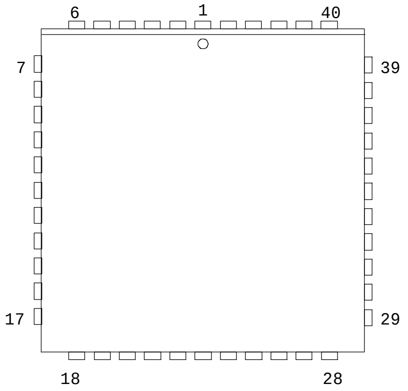


Figure 3. PLCC-44 Outline

CTSC9513A	x	x	-	x
Package				
Plastic DIP-40	P			
Plastic PLCC-44	J			
Temperature Range				
Industrial (-40 - 85º C)		I		
Maximum Clock Speed				
20 MHz				2

Table 2 - CTS9513 Ordering Information

GATE1-5 (Counter Gate Inputs)
Gate inputs are used to control counter behavior. Any gate may be routed to one of three internal counters. They may also be used as clock or count input sources for the internal counters or FOUT divider. The GATE lines may be programmed for use as counter enables, counter triggers or inhibits. Individual counters may be programmed for active polarity as well as to be level or edge sensitive to the GATE line.

OUT1-5 (Counter Outputs)
OUT1-5 are associated with individual counters. Outputs are tri-state and may be programmed by the counter for output polarity, initialized to a given state and programmed for pulse, square wave or complex duty cycle waveforms.

D0-15 (Data Bus)
D0-15 form a bi-directional 16 bit data bus for exchanging programming and status information with a host processor, or system. These lines act as inputs to the counter when CS and WR are asserted and as outputs when RD and CS are asserted. While CS is de-asserted these lines are placed in a high impedance state.

On power-up, the data bus is configured for 8 bit transfers. The data bus may be reconfigured for 16 bit by programming Master Mode register Bit 13. If D8-15 are not used they should be pulled up.

!CS (Chip Select Input)
The chip select line is an active low I/O control signal used to enable the device for read and write operations.

!WR (Write Input)
The write line is an active low I/O control signal which is used to transfer information from the data bus to one of the internal command or data registers.

!RD (Read Input)
The read line is an active low I/O control signal which is used to transfer information from one of the internal data or command registers to the data bus.

C/!D (Control/Data Port Select Input)
The C/D line is used in conjunction with the CS, RD, and WR to select which internal command or data register is being written to or read from. The C/D line selects between the command and data register sets as summarized in Table 3

FUNCTIONAL DESCRIPTION

SYSTEM LEVEL

The CTS9513 is addressed by the external system through two address locations. Counter and command data are written to individual counters through a sequence of indirectly addressing the internal command or data register through the command port address, followed by a write to the data port address which points to the indirectly addressed register location.

Data is transferred through either two 8 bit transfers or a single 16 bit transfer. Pointer sequencing for 8 bit transfers is automatic and is transferred as least significant byte first, most significant byte second.

Rapid programming of the CTS9513 may be accomplished by use of the auto-increment feature of the data pointer. This feature is enabled by setting Master Mode Register bit 14 (MM14). When enabled, the data pointer may be sequenced through a single counter group, all counter group registers, all counter group Hold registers only, or just the control group registers.

INTERNAL CONFIGURATION

Overview

A simplified block diagram of the CTS9513 is shown in Figure 4. This diagram shows the major device elements consisting of:

- five counter groups,
- internal frequency prescaler which divides down the primary external clock source from clock input X2,
- external FOUT clock prescalers which provide prescaled or divided outputs from a variety of sources,
- the Bus interface,
- Master mode register and
- the status register.

Not shown are the extended set registers, power-on reset circuitry or internal control lines. The counter group block diagrams are shown in Figures 5 and 6. Counter groups 1 and 2 as shown in Figure 5 have an additional programmable alarm register and 16 bit comparator for implementation of time-of-day and alarm functions.

Counter Groups

All of the counter groups have a 16 bit counter and four programmable registers. The primary and auxiliary counter mode register controls the count source, gating and counting modes, input and output polarities, binary or BCD counting and other parameters.

Load Register

The Load register is the primary register used for storing count-up or count-down values which may be automatically reloaded into the counter for repetitive counting.

Hold Register

The Hold register may be used for storing the instantaneous count value without disturbing the count process for reading by the host system. It may also be used in certain count modes for storing alternate count values and alternately counting the load and hold register values to generate complex waveforms.

Counter Outputs

Each of the counters has a single dedicated output pin which is programmable for polarity, tri-state, low-Z to ground and a variety of output modes as described later. This flexibility allows operation in a variety of bus and processor architectures.

Source Inputs

Each counter group may be programmed for a variety of count sources including any of the five source input lines, any of the internal prescaler outputs or the output of the previous counter, allowing counter concatenation and FOUT divided outputs.

Gate Inputs
Gate inputs are used for external hardware triggering or synchronization of the counters. Each counter may be programmed to be gated from its own gate line or the gate lines from the previous or next counter. The gate lines may also be programmed to be level or edge sensitive and respond to active high or low signals.

The gate line may be used to either initiate one or more count sequences or used as a count enable line, allowing the counter to count only while the gate line is held active. Another mode allows the counter to be reloaded from the load or hold register depending on the state of the gate line.

PROGRAMMING

REGISTER PROGRAMMING

Data Bus Operation
Table 3 summarizes the I/O control signal and data status during bus reads and writes to the CTS9513. The interface control logic assumes that

- RD and WR are never active simultaneously
- RD, WR, C/D are ignored unless CS is asserted.

Register Programming
Accessing and writing to a specific data or command register from the data port is as follows.

Set Data Pointer

1 Select the appropriate data pointer value to access the desired register (example Counter group 1 Mode register 0x01)
2 Write LOAD DATA POINTER command to primary command address (write 0x0001 to device address 0x01) to set data pointer to Counter Group 1 Mode register.

This points the data port to the Group 1 mode register and set the word pointer to 1 indicating a least significant word is expected.

WRITING TO REGISTERS

Write Data to Register

1 If the 16 bit transfer mode is selected, the next write to the Primary Data Port (Device Address 0x00) will write data to the Counter mode register.
2 If the 8 bit transfer mode is selected, the next write to the Primary Data Port Address will expect the least significant word of the register value, followed by a write of the most significant word to the data port. The internal word pointer is automatically incremented.
3 If an automatic sequence command has been given the data pointer will automatically be sequenced to the next register.

READING REGISTERS

Reading from a device register follows the write sequence very closely, requiring a write to the command register to set the appropriate data pointer, followed by a read or reads from the data port. Several items should be noted when reading from the device registers:

1 The data pointer should always be reloaded before reading from the data port if the prior command was anything but a LOAD DATA POINTER command in order to update the Read data pre-fetch latch.
2 A LOAD DATA POINTER command should be issued to the device prior to reading a HOLD register following a hardware triggered SAVE of the counter contents to the HOLD register.

COMMANDS

COUNTER COMMANDS

Counter commands are divided into two main groups. Those commands which directly affect counter operation, often shortcuts to programming specific register functions, and those associated with indirectly addressing the counters' internal registers.

Counter control commands can be further subdivided into those commands which affect individual counter operation and those which affect the overall device operation.

Table 4 Lists the commands associated with indirect addressing of the counter internal registers. These commands point the data port to the appropriate internal register in order to read or write to them.

Table 5 Lists the commands associated with controlling the actions of individual counters. They are made up basically of the ARM, DISARM, LOAD, SAVE, CLEAR, SET and STEP commands.

ARM Command
A counter must be ARMed before it can commence counting. Once ARMed, a counter may be programmed to begin counting immediately or to await a hardware trigger to initiate counting.

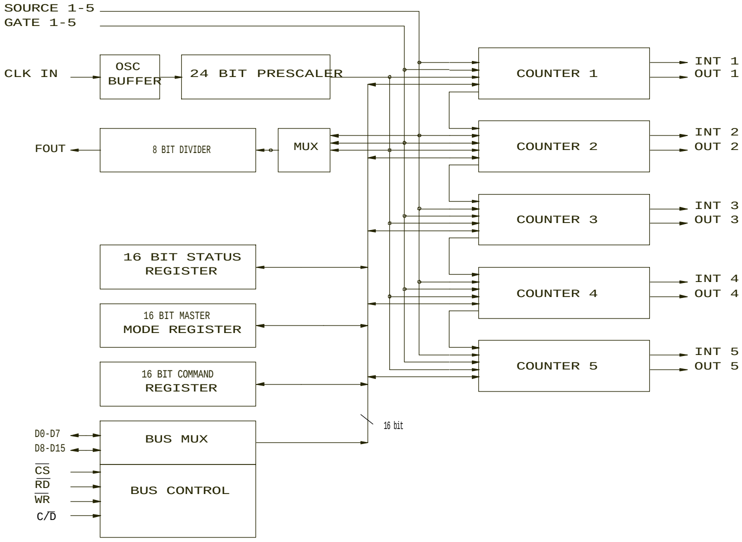


Figure 4 - CTS9513 Counter Block Diagram

DISARM Command
The DISARM command halts and disables any further counting regardless of any hardware gating or triggering. While DISARMed a counter may be reloaded, SAVEd or incremented or decremented using the STEP Command

LOAD Command
The LOAD command is used to load the counter with the value stored in either the associated Load or Hold register. It may also serve as an automatic retrigger of the counter once the counter is loaded.

SAVE Command
The SAVE command is used to save the contents of the counter while counting continues. This allows the counter value to be read without interfering with the counter. Subsequent SAVE commands will overwrite any previous contents of the Hold register.

CLEAR Command
The CLEAR command is used to reset the counter output toggle to initialize it to a low state. This command is only active if the output toggle is programmed. It is inactive if a Terminal Count output is specified.

SET Command
The SET command is used to set the counter output toggle to initialize it to a high state. This command is only active if the output toggle is programmed. It is inactive if a Terminal Count output is specified.

STEP Command
The STEP Command increments of decrements the selected counter by one depending on the operating mode.

Master Mode Commands
A number of commands directly affect the Master Mode Register without having to write to it directly. These commands affect primarily the modes of the data path, data pointer sequencing, enabling the divided FOUT output clocks and clearing of latched interrupt outputs from the counters. Table 6 summarizes these commands.

REGISTER DEFINITIONS

STATUS REGISTER

The 16 bit Status Register indicates the

1 Status of the internal word pointer
2 Status of the counter outputs
3 Status of the counter interrupt outputs

When reporting the status of the counter output, the status bit reflects the exact state of the output pin, regardless of how the output pin state or toggle is programmed.

CS	RD	WR	C/D	Dx
1	X	X	X	High Impedance
0	0	1	0	Read Data
0	0	1	1	Read Command
0	1	0	0	Write Data
0	1	0	1	Write Command
0	0	0	X	Illegal

Table 3 - CTS9513 Bus Control Line States

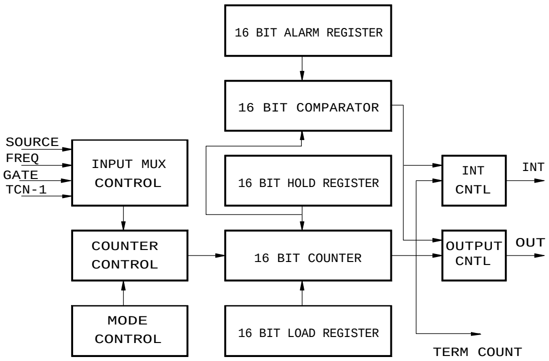


Figure 5 - CTS9513 Counter Groups 1 & 2

When an output low impedance to ground output is programmed, the Status bit reflects and Active High status. When the output is programmed for a high impedance output or is externally inhibited, the status register reflects an active low output. Table 7 summarizes the status register bit assignments.

Master Mode Commands
The Master Mode registers are 16 bit read/Write registers used to set counter parameters not associated with individual counters. These parameters include setting the data bus width, prescaling factors, Time of day functions and data pointer sequencing. The primary Master Mode Register is identical in function to the original '9513 device. The auxiliary Master Mode Register is used to program extended features of the CTS9513. If the auxiliary register is not programmed the device behaves as an original '9513 device. Table 8 summarizes the primary and auxiliary Master Mode Register bit assignments.

On Power-up the Master Mode register is cleared to all zeros resulting in the following default conditions:

1. Time of Day disabled
2. Alarm Comparators Disabled
3. FOUT source is F1
4. FOUT divider set for divide by 16
5. FOUT enabled
6. Data Bus 8 bits
7. Data Pointer Sequencing enabled
8. Frequency scaling Binary

Time of Day (Bits MM0-1)
Bits MM0 and MM1 control the Time-of-day functions for counters 1 and 2. When enabled, additional counter logic is enabled to allow the two counters to operate as a 24 hour clock.

Counters 1 and two must be programmed for BCD counting. To initialize the time, appropriate values are loaded in the Counter Load registers. To read the time a SAVE command is issued to Counters 1 and 2 and the values read from the Hold registers.

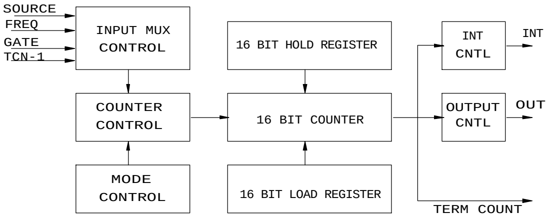


Figure 6 - CTS9513 Counter Groups 3 - 5

Table 9 illustrates the Time-of-day storage configuration. In short, Counter 2 bits 8-15 form a two digit BCD Hours counter, Bits 0-7 form a two digit BCD Minutes counter. Counter 1 bits Bits 8-15 form a two digit BCD seconds counter, Bits 4-7 form a tenth second counter and Bits 0-3 form a division factor for the input source for divide by 5, 6 or 10.

Comparator Enable (Bits MM2-3)
The two 16 bit comparators on counters 1 and 2 may be used in any mode. When enabled, the output of the comparators are routed to the output of the counter. The output will be asserted when the comparison between the counter and alarm register contents are true. It will remain asserted as long as the counter and alarm register remain the same.
In the Time-of-Day mode the comparators operate in conjunction such that the output of the counter 2 comparator is asserted only when both comparators 1 and 2 are true. the comparator 1 output will continue to operate normally.

FOUT Source (Bits MM4-7)
Fifteen different sources may be routed to the input of the FOUT divider, including the five SOURCE inputs, five GATE inputs and five of the internal divided frequencies derived from the X1 input. Additional Sources may be programmed using the extended Master mode register functions.

FOUT1 Divider (Bits MM8-11)
FOUT may be divided by 1 to 16. Master mode bits MM8-11 allow programming of the FOUT divider from 1 to 16 inclusive. Higher order division factors are programmed through the extended Master Mode register functions.

FOUT Enable (Bit MM12)
The FOUT output may be enabled or disabled and placed in a low impedance state to ground under software control.

Bus Width (Bit MM13)
When set, this bit places the device into a 16 bit external data bus mode. When cleared, the external data bus is set to 8 bits and registers are loaded 8 bits at a time, least significant word first.

Data Pointer Sequencing (Bit MM14)
When cleared, this bit enables automatic sequencing of the data pointer as defined by the data pointer commands. When set, the data pointer contents may only be changed by command.

Scaling (Bit MM15)
This bit determines whether the internal frequency prescaler operates as a BCD or Binary Divider. Figure 6 illustrates the internal 16 bit prescaler and its outputs.

COUNTER REGISTERS

Load Register
The load register is a read/write counter register used to store the counter initial value. The load register value can be transferred into the counter each time the counter reaches a “terminal count.” A “terminal count” is defined as that period of time the counter value would have been zero if an external value had not been transferred into the counter. In all operating modes the value in either the load or hold register is transferred into the counter when the counter reaches terminal count.

Hold Register
The hold register is a read /write dual purpose register. In some operating modes the hold register may be used to store counter instantaneous values on command without disturbing the counter action for readout by the host. Other operating modes allow the hold register to be used as storage for counter values in a fashion similar to the Load register. The counter may be loaded from the Hold register at terminal count, or alternately loaded from the Load and Hold register at terminal count.

Alarm Register
Counters 1 and 2 contain an additional 16 bit Alarm register and corresponding 16 bit comparator. When the value in the counter matches the value stored in the Alarm register the output pin for the counter goes true. The output remains true as long as the counter value matches the Alarm register value. The output may be programmed for active high or active low by the counter mode register.

COUNTER MODE REGISTER

Each counter group contains a mode control register which controls the counter behavior, gating and output active states and polarities and counter source. The counter mode register is initialized at power-up to all zeroes. This translates to an initial counter mode of:

1. Output Low impedance to Ground
2. Count Down
3. Count Binary
4. Count Once
5. Load Register Selected
6. No Retriggering
7. F1 source selected
8. Positive-true input polarity
9. No Gating

The Counter Mode Register must be loaded while the counter is disarmed.. Table 10 summarizes the Counter Mode Register bit assignments.

Output Control (Bits CM0-2)
The counter output may be configured to be disabled, programmed to follow the counter terminal count or to toggle its state at each terminal count. The output logic for each counter is shown in Figure 8.

The output may be disabled by either placing it in a high impedance state or in a low impedance state to ground. The outputs may also be hardware inhibited with the line.
In the Terminal count mode, the output may be programmed to output an active high or active low pulse which is equal to one count source clock period.

In the output toggle mode, the output changes state whenever the counter reaches a terminal count. The output state may be initialized with the SET and CLEAR counter commands.

C7	C6	C5	C4	C3	C2	C1	C0	Command Register Bit
0	0	0	E2	E1	G4	G2	G1	Load Data Pointer Commands
								G1-4 Group Pointer
								E1-2 Element Pointer
0	0	0	0	0	0	0	0	Reserved
0	0	0	0	0	0	0	1	Counter 1 Mode Register
0	0	0	0	0	0	1	0	Counter 2 Mode Register
0	0	0	0	0	0	1	1	Counter 3 Mode Register
0	0	0	0	0	1	0	0	Counter 4 Mode Register
0	0	0	0	0	1	0	1	Counter 5 Mode Register
0	0	0	0	0	1	1	0	Reserved
0	0	0	0	0	1	1	1	Alarm Register 1 / Control Cycle
0	0	0	0	1	0	0	0	Reserved
0	0	0	0	1	0	0	1	Counter 1 Load Register
0	0	0	0	1	0	1	0	Counter 2 Load Register
0	0	0	0	1	0	1	1	Counter 3 Load Register
0	0	0	0	1	1	0	0	Counter 4 Load Register
0	0	0	0	1	1	0	1	Counter 5 Load Register
0	0	0	0	1	1	1	0	Reserved
0	0	0	0	1	1	1	1	Alarm Register 2 / Control Cycle
0	0	0	1	0	0	0	0	Reserved
0	0	0	1	0	0	0	1	Counter 1 Hold Register
0	0	0	1	0	0	1	0	Counter 2 Hold Register
0	0	0	1	0	0	1	1	Counter 3 Hold Register
0	0	0	1	0	1	0	0	Counter 4 Hold Register
0	0	0	1	0	1	0	1	Counter 5 Hold Register
0	0	0	1	0	1	1	0	Reserved
0	0	0	1	0	1	1	1	Master Mode Register / Control Cycle
0	0	0	1	1	0	0	0	Reserved
0	0	0	1	1	0	0	1	Hold Register Cycle
0	0	0	1	1	0	1	0	Hold Register Cycle
0	0	0	1	1	0	1	1	Hold Register Cycle
0	0	0	1	1	1	0	0	Hold Register Cycle
0	0	0	1	1	1	0	1	Hold Register Cycle
0	0	0	1	1	1	1	0	Reserved
0	0	0	1	1	1	1	1	Status Register

Table 4 - CTS9513 Data Pointer Commands

Count Control (Bits CM3-7)

Whenever the counter reaches a TC, the counter automatically reloads the counter from the Load or Hold Register. Which register the counter loads from, whether the counter counts repeatedly or once, whether the counter counts binary or BCD and whether the counter is under hardware control is controlled by the Count control.

Bit CM3 controls whether the counter counts in Binary or BCD fashion. Bit CM4 determines whether the counter counts up or down. Bit CM5 determines whether the counter counts once and disarms itself, or will continue counting and reloading the counter until commanded to disarm.
Bit CM6 determines the source from which the counter will be reloaded. The actions of CM6 depend on the gating control settings. If CM6 is cleared, the counter reloads from the Load Register at TC. If CM6 is set, the counter may reload from either the Load or the hold register depending on the gating mode. It may alternate with the Load register or be controlled from the gate to reload from the load or hold register.

Bit CM7 controls whether hardware retriggering of the counter is enabled. Its actions depend on the settings of CM5, CM6 and the gating controls.

If some type of gating is enabled and CM7 is cleared, hardware retriggering is disabled. When CM7 is set, hardware retriggering is enabled and the counter is retriggered any time an active gate edge is received. When retriggered the counter value is saved in the Hold register and the counter reloaded from the Load register.

If no gating is enabled and CM7 is cleared, the gate input has no effect on counting. If CM7 is set then the Gate input controls whether the counter is reloaded from the Load or Hold Register.

C7	C6	C5	C4	C3	C2	C1	C0	Command Register Bit
			S5	**S4**	**S3**	**S2**	**S1**	**S1-5 - Counter Group Select**
0	0	1	S5	S4	S3	S2	S1	Arm Selected Counters
0	1	0	S5	S4	S3	S2	S1	Load Selected Counters
0	1	1	S5	S4	S3	S2	S1	Load and Arm Selected Counters
1	0	0	S5	S4	S3	S2	S1	Disarm and Save Selected Counters
1	0	1	S5	S4	S3	S2	S1	Save selected counters to Hold Registers
1	1	0	S5	S4	S3	S2	S1	Disarm Selected Counters
					N4	**N2**	**N1**	**N1-4 Counter Group Select (001 = N = 101**
1	1	1	0	0	N4	N2	N1	Clear Selected Counter Toggle Out
1	1	1	0	1	N4	N2	N1	Set Selected Counter Toggle Out
1	1	1	1	0	N4	N2	N1	Step Selected Counter (up/down by CM3)

Table 5 - Counter Action Related Commands

Count Source (Bits CM8-12)

The count source determines which source is used as an input to the counter. There are 20 possible count sources, 16 of which may be selected with bits CM8-12. Additional Count sources may be specified with the extended registers. Figure 8 illustrates the internal 24 bit prescaler whose outputs may be used as count sources.

Gating Control (Bits CM13-15)

Gating control determines whether the counter is hardware gated or not. When gating is disabled the counter will continue as long as the counter is armed. If any gating mode is enabled the counter action is determined by some hardware gate condition.

Gating of the counter may be controlled from the gate line associated with the counter or gate lines associated with adjacent counters. Gating on the line associated with the counter may be programmed for edge or level sensitive, active high or active low. The counter may also be gated by the TC output of the previous counter. The gating control logic is outlined in Figure 7.

COUNTER MODES

Counter modes continue as in the ‘9513 to retain their mode designations A-X, with modes M, P, T, U and V reserved. Tables 11-12 summarize the counter modes and the associated settings of the counter mode bits CM5-7 and CM13-15.

Figures 10 through 28 illustrate the counter modes. All representative waveforms assume counting down on rising source edges. A TC mode and Toggled output waveform are shown in each waveform. For waveforms which disarm automatically on TC the software ARM command is shown in conjunction with

C7	C6	C5	C4	C3	C2	C1	C0	Command Register Bit
1	1	1	0	0	0	0	0	Clear MM14 (Enable Data Pointer Sequencing)
1	1	1	0	0	1	1	0	Clear MM12 (FOUT Gate On)
1	1	1	0	0	1	1	1	Clear MM13 (Enable 8 bit Bus Mode)
1	1	1	0	1	0	0	0	Set MM14 (Disable Data Pointer Sequencing)
1	1	1	0	1	1	1	0	Set MM12 (FOUT Gate Off)
1	1	1	0	1	1	1	1	Set MM13 (Enable 16 bit Bus Mode)
1	1	1	1	0	0	0	0	**(Originally Reserved)**
1	1	1	1	0	1	1	0	**(Originally Reserved)**
1	1	1	1	0	1	1	1	**(Orig Reserved)**
1	1	1	1	1	0	0	0	Enable Write Pre-Fetch
1	1	1	1	1	0	0	1	Disable Write Pre-Fetch
1	1	1	1	1	0	1	0	**(Orig Reserved)**
1	1	1	1	1	0	1	1	**(Orig Reserved)**
1	1	1	1	1	1	0	0	**(Orig Reserved)**
1	1	1	1	1	1	0	1	**(Orig Reserved)**
1	1	1	1	1	1	1	0	**(Orig Reserved)**
1	1	1	1	1	1	1	1	Master Reset

Table 6 - Device Level Commands

S7	S6	S5	S4	S3	S2	S1	S0
CMP2	CMP1	OUT5	OUT4	OUT3	OUT2	OUT1	WP
Comparator Reflects actual state of Interrupt Output		Counter Output Status Reflects Actual State of Output					Byte Pointer

Table 7 - Status Register

a Write pulse. Repetitive waveforms do not show the write pulse or ARM command. The letters L and H are used in the figures to denote Load and Hold register values and the letters K and N to denote arbitrary counter values.

In all cases, the counter begins counting on the first count source edge following the Write pulse in software triggered modes and the first source edge following a valid gate edge in hardware triggered or enabled modes.

In gate controlled modes which inhibit counting, the counter is suspended for any valid source edges that occur after de-assertion of the gate line.

CTS9513AXI-2 ERRATA

Although tested extensively to ensure full compliance with the original AM9513Axx device functions and operating modes, several functional anomalies have come to our attention. Both current and potential users of this device should take note of these.

Devices Affected:
All 1996, 97, 98, 99 and 2000 devices manufactured to date

Planned Action:
There are no immediate plans to correct these defects until further testing can be completed to detect any further anomalies.

Work-Arounds:
There is no current work-around for these problems for existing designs.

CRYSTAL OSCILLATOR

The CTS9513 does not incorporate a crystal oscillator and must be driven from an external TTL compatible oscillator source.

COMMAND DATA READ/WRITE DATA LATCH

In this implementation of the 9513 data being written to the device is not latched on the rising (trailing) edge of the write strobe. Data in this device is latched into the command and control registers on the low level of the write strobe. This means that the data must be stable up until shortly before the rising edge of the write strobe. This appears to be an artifact of the way the 8 bit sequential write mode was implemented in order to correctly increment the byte pointer and latch the data on the one write strobe.

To date we have only seen this create a problem in one instance on an ISA bus Counter/Timer instrumentation card where the ISA bus decoding was incorrectly implemented. In that instance, a delay in de-asserting the chip select was causing the leading edge of a write strobe for another I/O device to appear prior to the trailing edge of the Chip Select signal. This was interpreted as another valid write to the 9513 device causing invalid data to be written to the device.

MM15	MM14	MM13	MM12	MM11	MM10	MM9	MM8	MM7	MM6	MM5	MM4	MM3	MM2	MM1	MM0
SCALE	POINT	BUS	FGATE1	DIV1-8	DIV1-4	DIV1-2	DIV1-1	FOUT1-8	FOUT1-4	FOUT1-2	FOUT1-1	COMP2	COMP1	TOD2	TOD1
Scale Mode 0 BIN 1 BCD	Data Pointer 0 Enable 1 Disable	Data Bus Width 0 = 8 1 = 16	FOUT Mode 0 = On 1 = Off	FOUT Divider 0000 = Divide by 16 0001 = Divide by 1 0010 = Divide by 2 0011 = Divide by 3 0100 = Divide by 4 0101 = Divide by 5 . . . . 1111 = Divide by 16				FOUT Source Select 0000 = F1 001 = Source 1 0010 = Source 2 0011 = Source 3 0100 = Source 4 0101 = Source 5 0110 = Gate 1 0111 = Gate 2 1000 = Gate 3 1001 = Gate 4 1010 = Gate 5 1011 = F1 1100 = F2 1101 = F3 1110 = F4 1111 = F5				Comparator Mode 00 = Disabled 01 = Comparator 1 On 10 = Comparator 2 On 11 = Both On		Time of Day Mode 00 = TOD Disabled 01 = TOD Enabled /5 10 = TOD Enabled /6 11 = TOD Enabled /10	

Table 8 - Master and Auxiliary Master Mode Register Definitions

SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE

C2-15	C2-14	C2-13	C2-12	C2-11	C2-10	C2-9	C2-8	C2-7	C2-6	C2-5	C2-4	C2-3	C2-2	C2-1	C2-0
10's Hours				Hours				10's Minutes				Minutes			
BCD DATA 0 - 23 Hours								BCD DATA 0-59 Minutes							

C1-15	C1-14	C1-13	C1-12	C1-11	C1-10	C1-9	C1-8	C1-7	C1-6	C1-5	C1-4	C1-3	C1-2	C1-1	C1-0
10's Seconds				Seconds				10th Seconds				Division Factor (5, 6, 10)			
BCD DATA 0.0 - 59.9 Seconds															

Table 9 - CTS9513 Time-of-Day Data Format

CM15	CM14	CM13	CM12	CM11	CM10	CM9	CM8	CM7	CM6	CM5	CM4	CM3	CM2	CM1	CM0
GCTL3	GCTL2	GCTL1	EDGE	SRC1-8	SRC1-4	SRC1-2	SRC1-1	GATE	RELOAD	REPEAT	COUNT	DIR	OUT4	OUT2	OUT1
Gate Control 000 No Gating 001 Active High, TC N-1 010 Active High Level GateN+1 011 Active High Level GateN-1 100 Active High Level GateN 101 Active Low Level GateN 110 Active High Edge GateN 111 Active Low Edge GateN			**Edge Mode** 0 Rising 1 Falling	**Count Source Selection** 0000 = TC N-1 0001 = Source 1 0010 = Source 2 0011 = Source 3 0100 = Source 4 0101 = Source 5 0110 = Gate 1 0111 = Gate 2 1000 = Gate 3 1001 = Gate 4 1010 = Gate 5 1011 = F1 1100 = F2 1101 = F3 1110 = F4 1111 = F5				**Gate Mode** 0 = Off 1 = On	**Reload Mode** 0 = Load 1 = Both	**Repeat Mode** 0 Once 1 Repeat	**Count Mode** 0 Binary 1 BCD	**Count Direction** 0 Down 1 Up	**Output Control** 000 = Inactive, Output Low 001 = Active High on TC 010 = TC Toggled 011 = Illegal 100 = Inactive, Output High Z 101 = Active Low on TC 110 = Illegal "111 = Illegal		

Table 10 - CTS9513 Counter Mode and Auxiliary Counter Mode Register Bit Assignments

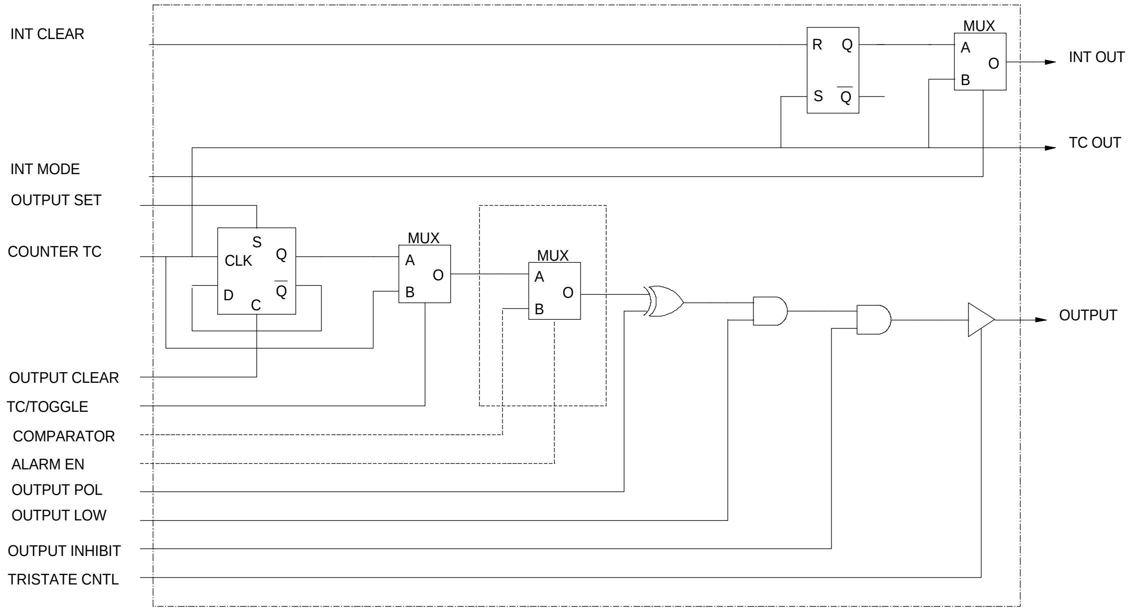
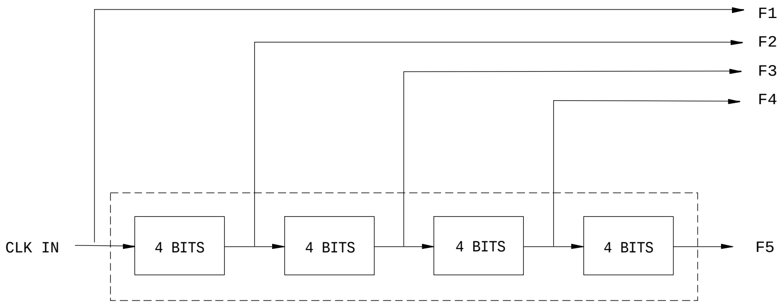


Figure 7 - Counter Output Section Block Diagram



FREQUENCY	BCD SCALING	BINARY SCALING
F1	CLK IN	CLK IN
F2	CLK / 10	CLK / 16
F3	CLK / 100	CLK / 256
F4	CLK / 1000	CLK / 4096
F5	CLK / 10000	CLK / 65536

Figure 8 - CTS9513 Counter Internal Prescaler Block Diagram

MODE V (FSK) ERROR

An error in implementing the special gate function prevents the implementation of Mode V (FSK Generator). The gate level is supposed to control whether the counter is reloaded from the LOAD or HOLD register to determine the output rate generator frequency and allow switching between two frequencies to produce Frequency Shift Keying (FSK) modulation.

When programmed for Mode V, the current device Revision will reload only from the HOLD register regardless of the state of the GATE input.
This appears to be a general problem with the “special gate” function that controls reloading of the counter from the Load or Hold register depending on the state of the gate.

MODE J ERROR

The counters will not allow a count of 1 to be set in the load and/or hold registers

COUNTER SAVE ERRORS

Due to the asynchronous nature of this part (and to an extent the original AMD AM9513) we have seen errors in the saved counter data when a counter save command is issued. This occurs when the write strobe rising edge for a save command occurs simultaneously with a counter clock edge and the counter tries to save the current count while also trying to increment or decrement the counter.

The only solid solution we have found for this proble is for the bus clock to also be the master clock or to be phased locked to it in order for the timing of bus read/write cycles to be deterministic with respect to the counter clock edges.

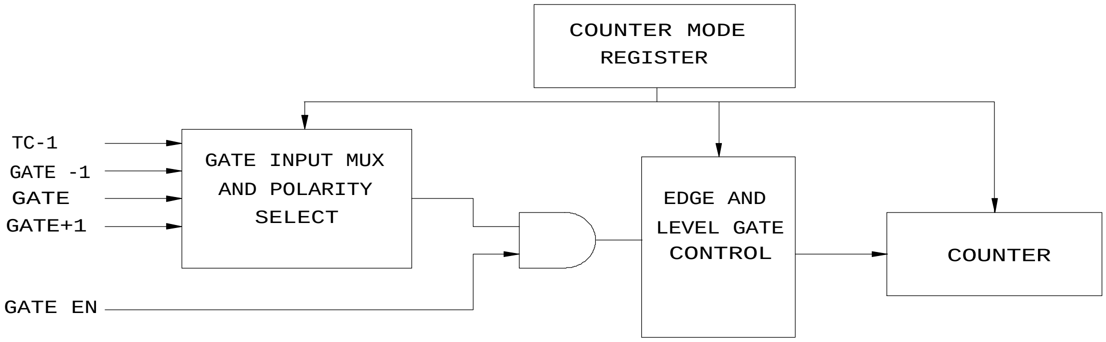


Figure 9 - CTS9513 Counter Gating Input Logic Block Diagram

OPERATING MODE	A	B	C	D	E	F	G	H	I	J	K	L
CM7 (SPECIAL GATE)	0	0	0	0	0	0	0	0	0	0	0	0
CM6 (RELOAD SOURCE)	0	0	0	0	0	0	1	1	1	1	1	1
CM5 (REPITITION)	0	0	0	1	1	1	0	0	0	1	1	1
CM13-15 (GATE CONTROL)	000	LVL	EDG	000	LVL	EDG	000	LVL	EGD	000	LVL	EDG
Count to TC Once	X	X	X									
Count to TC Twice							X	X	X			
Count to TC repeatedly				X	X	X				X	X	X
Gate Input Inactive	x			x			x			x		
Count while gate active		x			x			x			x	
Count once on gate edge			x			x						
Count twice on gate edge									x			x
No Hardware retriggering	x	x	x	x	x	x	x	x	x	x	x	x
Reload from Load on TC	x	x	x	x	x	x						
Alternate Load/Hold on TC							x	x	x	x	x	x
Gate Controlled Load/Hold												
Gate Retrigger Counter												

Table 11 - Counter Modes A-L

OPERATING MODE	M	N	O	P	Q	R	S	T	U	V	W	X
CM7 (SPECIAL GATE)	1	1	1	1	1	1	1	1	1	1	1	1
CM6 (RELOAD SOURCE)	0	0	0	0	0	0	1	1	1	1	1	1
CM5 (REPITITION)	0	0	0	1	1	1	0	0	0	1	1	1
CM13-15 (GATE CONTROL)	000	LVL	EDG	000	LVL	EDG	000	LVL	EGD	000	LVL	EDG
Count to TC Once		X	X									
Count to TC Twice							X					
Count to TC repeatedly					X	X				X		
Gate Input Inactive							X			X		
Count while gate active		X			X							
Count once on gate edge			X			X						
Count twice on gate edge												
No Hardware retriggering							X			X		
Reload from Load on TC		X	X		X	X						
Alternate Load/Hold on TC												
Gate Controlled Load/Hold							X			X		
Gate Retrigger Counter		X	X		X	X						

Table 12 - Counter Modes M-X

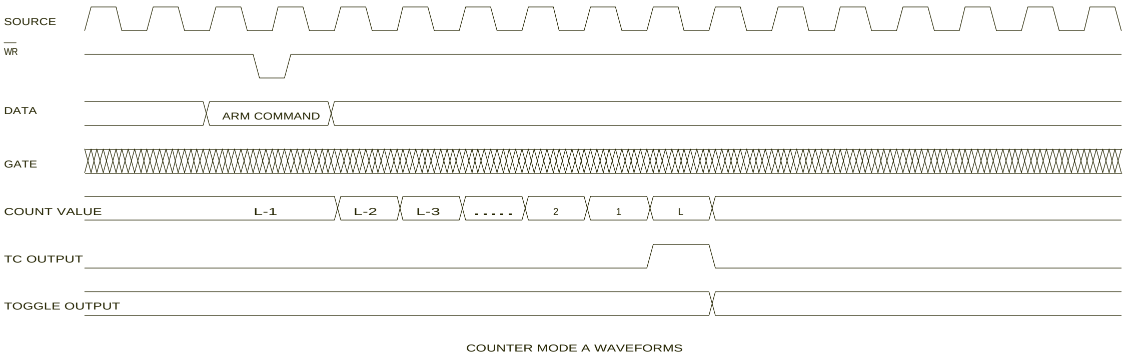


Figure 10 - CTS9513 Counter Mode A Representative Waveforms

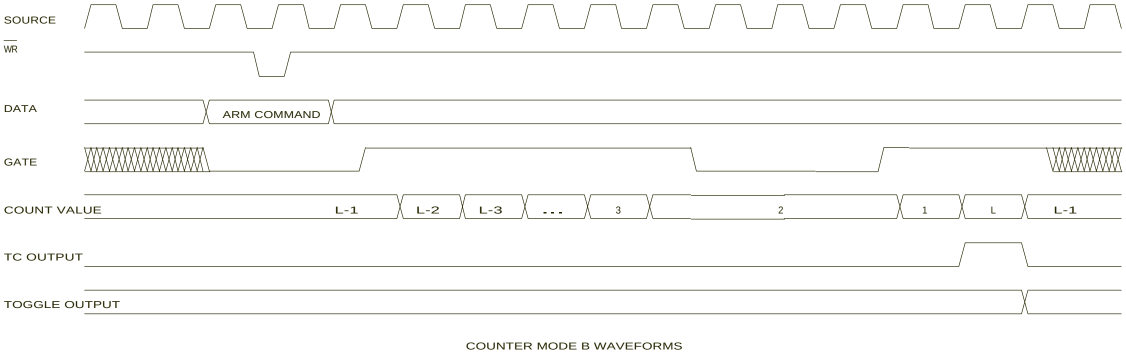


Figure 11 - CTS9513 Counter Mode B Representative Waveforms

Mode A - Software Triggered Strobe with no Gating
As shown in Figure 10, The counter is only active after receipt of an ARM command. On reaching TC the counter automatically reloads from the Load register and disarms, awaiting the next software ARM command.

MODE B - SOFTWARE TRIGGERED STROBE WITH LEVEL GATING

In Mode B, illustrated in Figure 11 the counter is only active when both an ARM command has been received and the selected Gate line is active. The counter will halt counting when the gate line is de-asserted and resume counting when the gate line is re-asserted until the counter reaches TC. When the counter reaches TC the timer will reload from the load register and disarm automatically until a new ARM command is received.

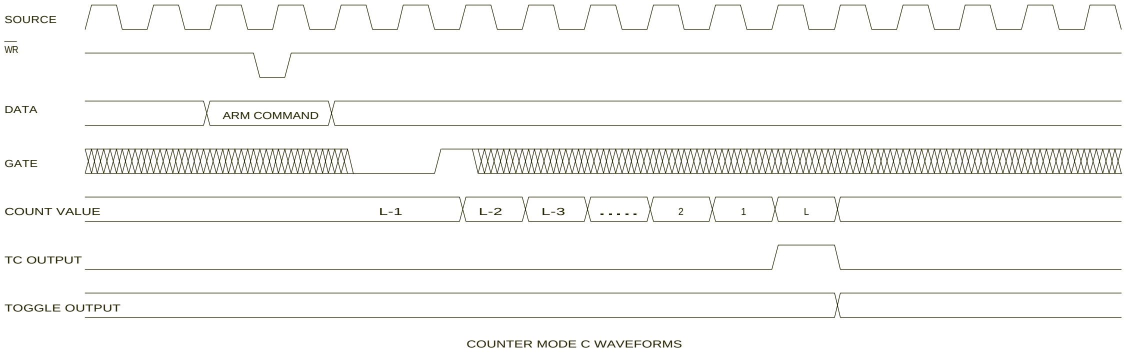


Figure 12 - CTS9513 Counter Mode C Representative Waveforms

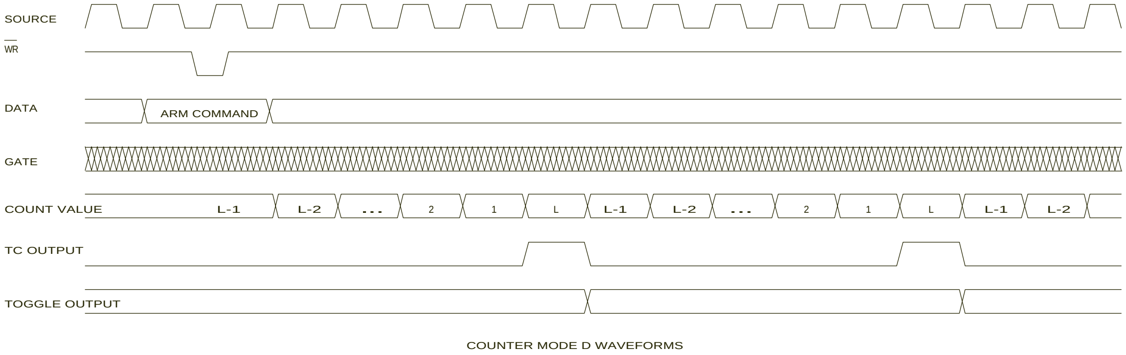


Figure 13 - CTS9513 Counter Mode D Representative Waveforms

MODE C HARDWARE TRIGGERED STROBE

In Mode C, as shown in Figure 12, the counter is active only after receipt of an ARM command and the application of a Gate edge to the selected gate line. Once a Gate edge is sensed, the counter will count until it reaches TC. Subsequent gate actions have no further effect on the counter action. The counter will remain inactive until receipt of a new ARM command and Gate.

MODE D RATE GENERATOR WITH NO HARDWARE GATING

Mode D, illustrated in Figure 13. is commonly used as a programmable frequency source as it continues to count repetitively until receipt of a DISARM command. Once ARMed, the counter counts to TC, automatically reloads the counter from the Load register and begins counting again. The waveform produced can be a square wave if the Toggle output mode is specified. The Gate line has no effect on the counter action.

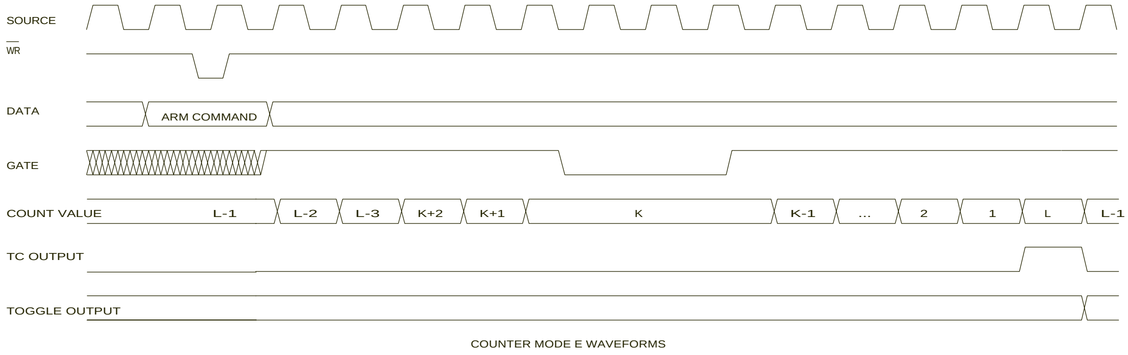


Figure 14 - CTS9513 Counter Mode E Representative Waveforms

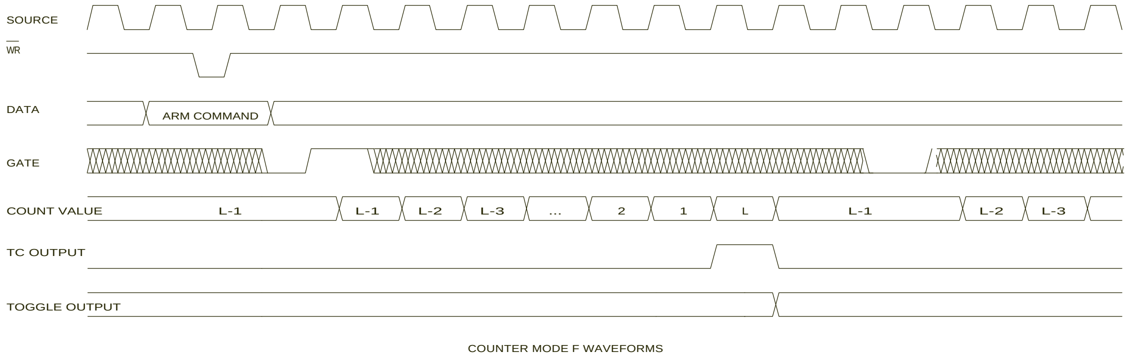


Figure 15 - CTS9513 Counter Mode F Representative Waveforms

MODE E RATE GENERATOR WITH LEVEL GATING

Mode E is similar to Mode D in that the counter will count repetitively after being ARMed and as long as the selected Gate line is asserted. As shown in Figure 14, this allows gating of the pulse train or square wave on and off from an external source via the gate line.

MODE F NON-RETRIGGERABLE ONE SHOT

Mode F is similar to Mode C with the exception that the counter may be retriggered without receipt of a new ARM Command. As shown in Figure 15, Once the counter has been ARMed, and a valid Gate edge has been received, the counter will count once to TC and reload the counter from the Load register. It will remain inactive until receipt of another Gate edge. While counting, subsequent gate edges are disregarded.

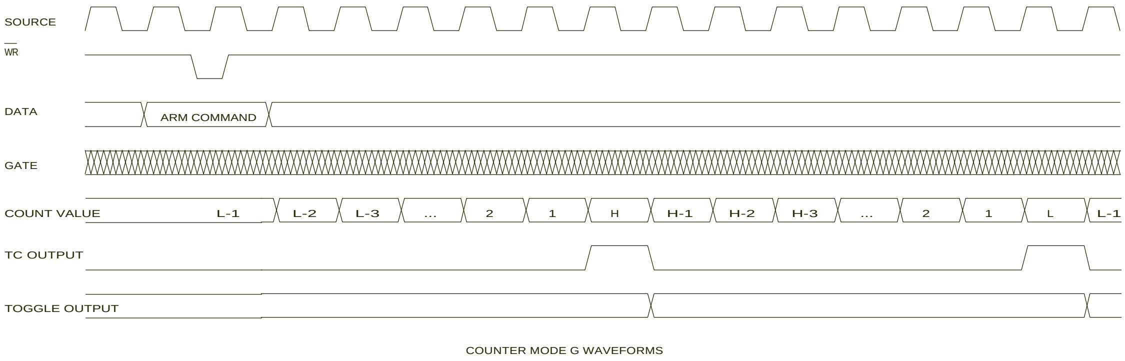


Figure 16 - CTS9513 Counter Mode G Representative Waveforms

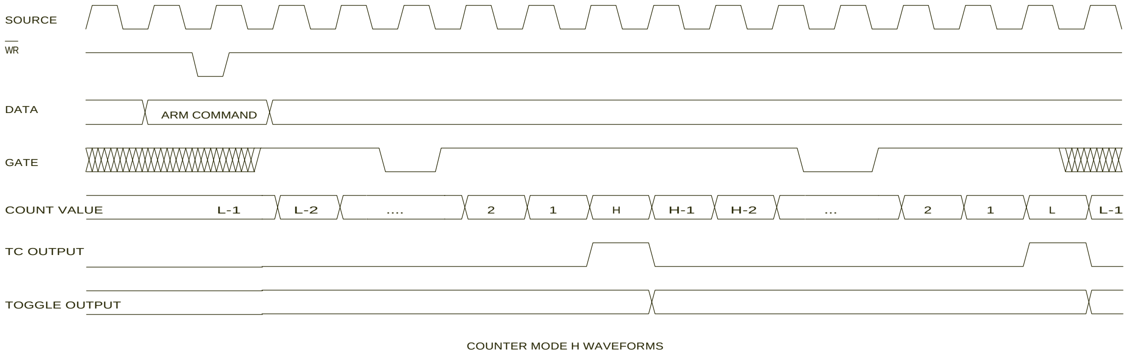


Figure 17 - CTS9513 Counter Mode H Representative Waveforms

Mode G Software Triggered Delayed Pulse One-Shot

In Mode G, once the counter has been ARMed, the counter will:

1 Count to TC with the Load register value
2 Reload itself automatically from the Hold Register.
3 Count to TC with the Hold Register Value
4 Disarm itself and reload the counter with the Load register Value.

This produces a waveform as illustrated in Figure 16 in which the counter can in TC mode produce a pair of pulses with the first pulse delay controlled by the Load count value and the delay between the pulses determined by the Hold register count.

If the Toggle Output mode is selected, the output produced is a pulse width determined by the Hold count and an initial delay determined by the Load count. This is the more common use of this mode of operation.

Mode H Software Triggered Delayed Pulse One-Shot with Hardware Gating

Mode H is similar to Mode G with the exception that the counter is active only after receipt of an ARM command and a valid Gate input. As shown in Figure 17 the counter counts only as long as the Gate line is asserted and suspended while the Gate line is de-asserted. Tas in Mode G the counter counts to TC using the Load register value, reloads from the hold register and counts to a second TC. Once the counter reaches the second TC the counter disarms itself and awaits another ARM command.

This mode allows extension of either the initial delay or the delayed pulse width by use of the Gate.

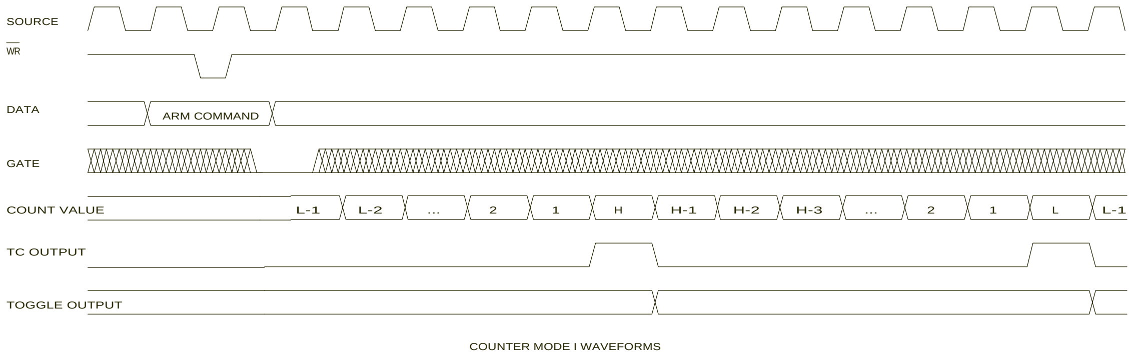


Figure 18 - CTS9513 Counter Mode **I** Representative Waveforms

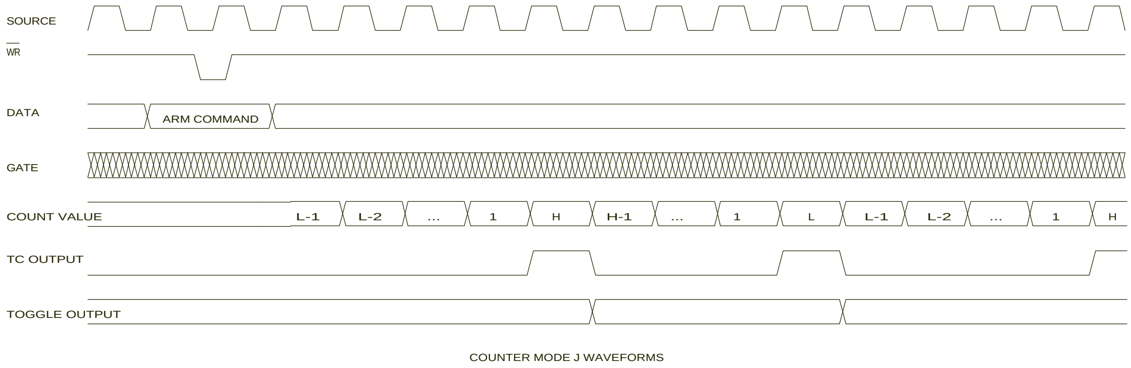


Figure 19 - CTS9513 Counter Mode J Representative Waveforms

MODE I HARDWARE TRIGGERED DELAYED PULSE STROBE

Mode I is similar to Mode G with the exception that the counter is active only after receipt of an ARM command and a valid Gate Edge. As illustrated in Figure 18, the counter will count to TC, reload from the Hold Register, count to TC then disarm itself. Once a valid Gate edge has been received the gate line has no further action on the counter.

MODE J VARIABLE DUTY CYCLE RATE GENERATOR WITH NO HARDWARE GATING

This mode is used primarily for generation of variable duty cycle waveforms. Once armed the counter will count repeatedly until disarmed. The counter will count to the first TC, reload automatically from the Hold register, count to the next TC, reload automatically from the Load register and repeat the cycle. If the toggle output mode is selected, the output will have an on(or off) time equal to the load count and off(on) time equal to the hold count. As shown in Figure 19.

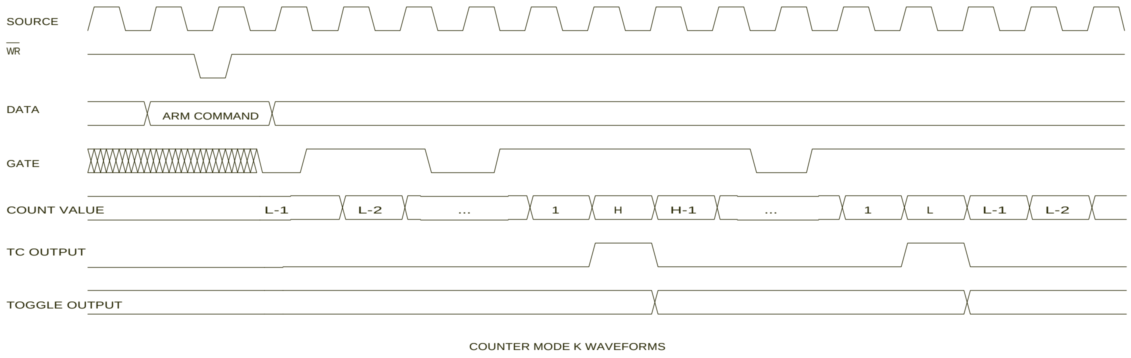


Figure 20 - CTS9513 Counter Mode K Representative Waveforms

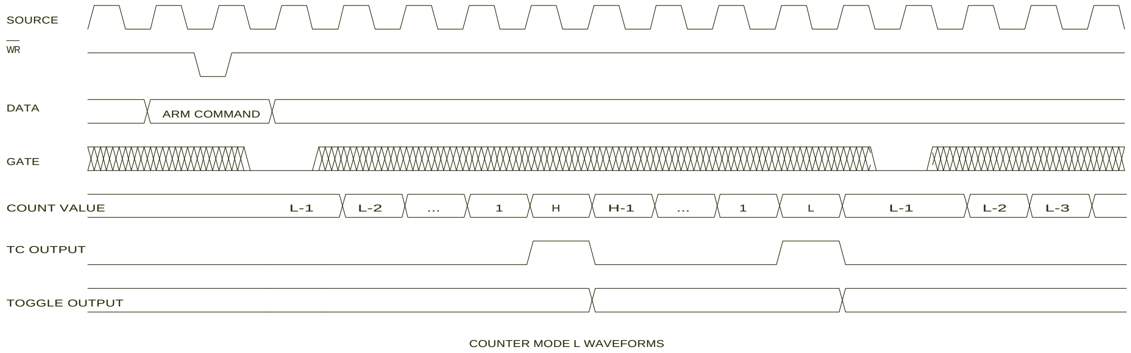


Figure 21 - CTS9513 Counter Mode L Representative Waveforms

MODE K VARIABLE DUTY CYCLE RATE GENERATOR WITH LEVEL GATING

Mode K is similar to Mode J with the exception that the counter is enabled only after being ARMed and when the selected Gate line is asserted. When the Gate line is deasserted the counter stops. This allows the gate to modulate the duty cycle of either state as illustrated in Figure 20.

MODE L HARDWARE TRIGGERED DELAYED PULSE ONE-SHOT

Mode L is used often as an externally triggered delayed pulse generator, where the delay and pulse width are both programmable. Like Modes J and K, the counter cycles through the load count, reloads from the hold at the first TC, and counts to the second TC.

Unlike Modes J and K, however the counter is only active after being ARMed and after a valid gate edge is received. As shown in Figure 21 the gate edge initiates one count cycle and is disregarded for the rest of the cycle. After one count cycle (Load and Hold) the counter stops until another gate edge is received.

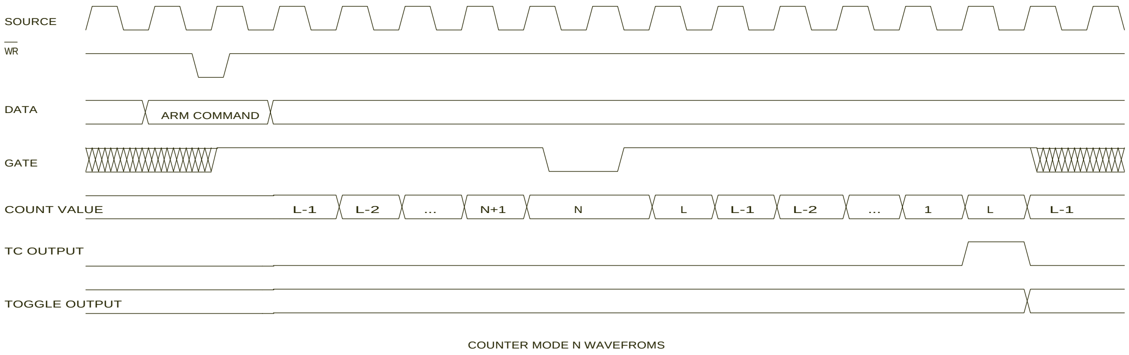


Figure 22 - CTS9513 Counter Mode N Representative Waveforms

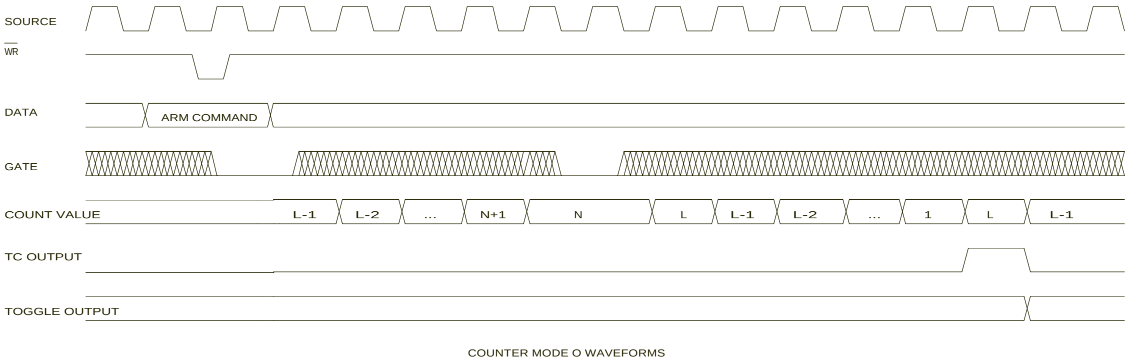


Figure 23 - CTS9513 Counter Mode O Representative Waveforms

MODE N SOFTWARE TRIGGERED STROBE WITH LEVEL GATING AND HARDWARE RETRIGGERING

In Mode N, once ARMed, the counter is active only as long at the selected Gate line is asserted. Counting begins only after the gate line is asserted after the counter is ARMed. If the Gate line remains asserted the counter will count to TC, reload automatically from the load register and disarm itself until receipt of a new ARM command. If the gate is deasserted prior to the counter reaching TC the counter will halt. When the Gate line is reasserted on a halted counter, the count value is transferred to the Hold register and the next valid count source edge will cause the counter to reload from the Load register and begin counting again, effectively retriggering the counter as shown in Figure 22.

One application of this mode is to measure the delay between two successive gate edges by reading the remainder count value from the hold register.

MODE O SOFTWARE TRIGGERED STROBE WITH EDGE GATING AND HARDWARE RETRIGGERING

Mode O is similar to Mode N in that the counter must be ARMed and a valid Gate edge must be received to start the counter. Unlike most other modes, however, each time a valid gate edge is received prior to the counter reaching TC will cause the counter to be retriggered by reloading the counter from the load register on the first valid source edge following a valid gate edge. If the counter is allowed to reach TC is automatically reloads from the Load register and disarms itself.

The counter is insensitive to gate edges while disarmed and while counting. The counter is sensitive only to a valid gate edge while counting.

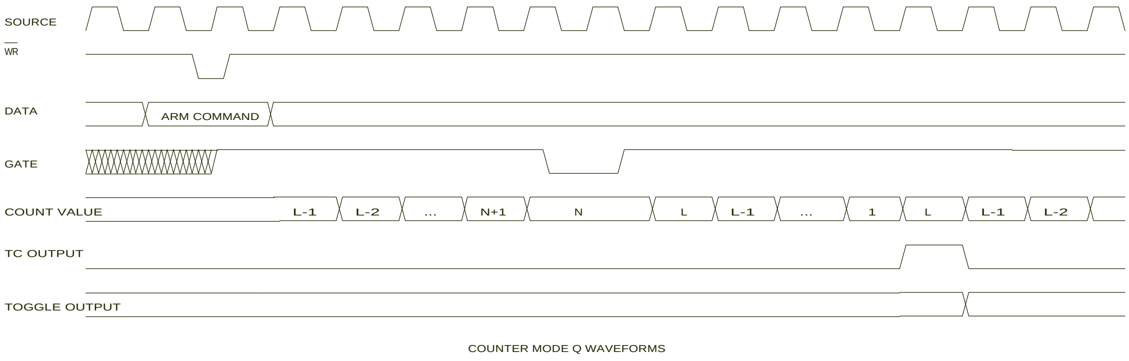


Figure 24 - CTS9513 Counter Mode Q Representative Waveforms

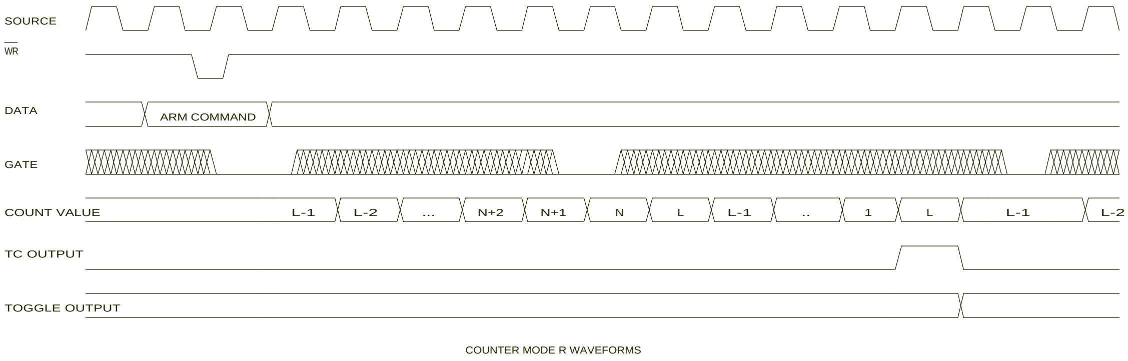


Figure 25 - CTS9513 Counter Mode R Representative Waveforms

MODE Q RATE GENERATOR WITH SYNCHRONIZATION

Mode Q provides a continuous rate generator which may be externally gated or synchronized to an external event via the Gate input. As shown in Figure 24, once an ARM command is received, the counter will continuously count to TC, reload the Load register and repeat as long as the Gate line is asserted. While the Gate line is deasserted the counter is inhibited. On the active going edge of the gate signal the counter is reloaded from the Load register, resetting the counter and resume counting on the second valid source edge following the Gate edge.

MODE R RETRIGGERABLE ONE-SHOT

Mode R, as shown in Figure 25, begins counting only after receipt of an ARM command and a valid active Gate edge. The counter will count once to TC and stop. The counter will remain inactive until receipt of a subsequent valid Gate edge.

If a valid Gate Edge is received prior to the counter reaching TC the counter value will be saved in the Hold register and the counter reloaded from the Load register, retriggering or resetting the counter. The counter in insensitive to the Gate level and gate actions do no inhibit the counter as in Mode Q.

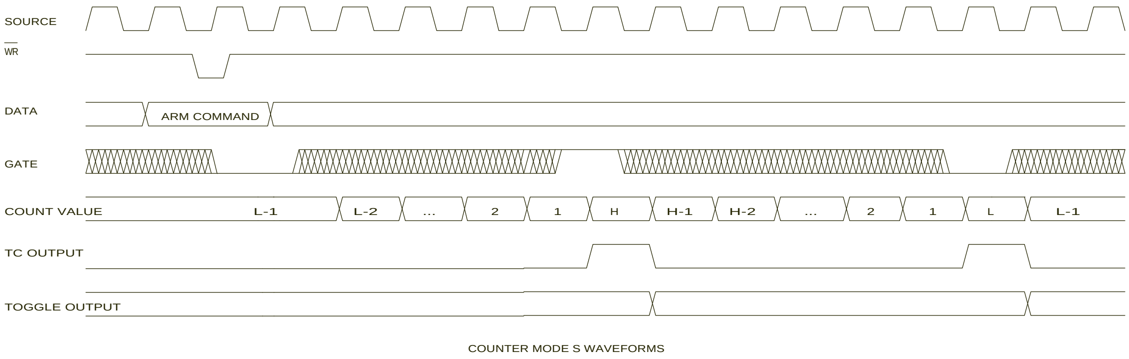


Figure 26 - CTS9513 Counter Mode S Representative Waveforms

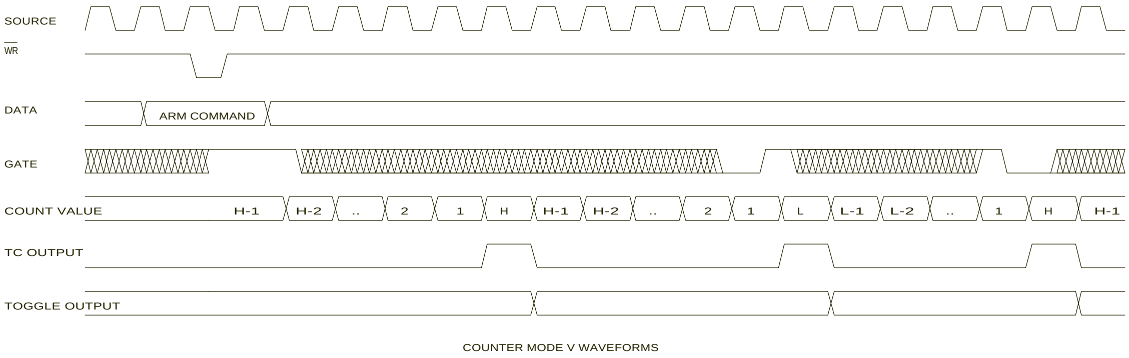


Figure 27 - CTS9513 Counter Mode V Representative Waveforms

MODE S GATE CONTROLLED STROBE

In Mode S, once ARMed the counter will count to TC twice and disarm. During this time the State of the Gate line determines whether the counter is loaded from the Load or Hold Register. The Gate line does not affect or initiate the counter in this Mode. Its only action is a level sensitive selection of the Load or Hold Register as a counter reload source.

As shown in Figure 26, at each TC in the cycle, if the Gate line is high, the counter will be reloaded from the Hold Register. If it is Low the counter is reloaded from the Load Register.

MODE V FREQUENCY SHIFT KEYING

Mode V is similar to mode S in that the Gate line act to select which register the counter is reloaded from, but counts continuously once armed. If the Toggled output is used, the output may be used to switch between two frequencies determined by the Load and Hold Count values and the state of the Gate line as shown in Figure 27. This is used in Frequency Shift Keying (FSK) applications.

NOTE: This mode does not function correctly in current devices. Please see the ERRATA section for more information

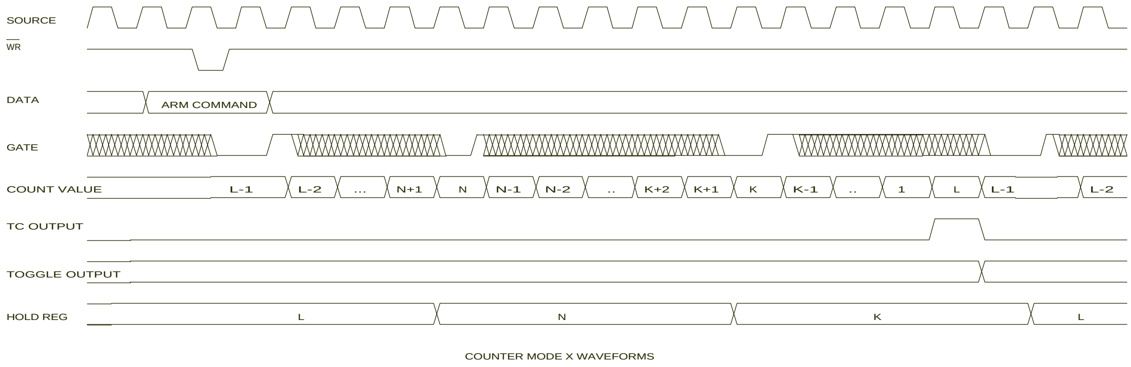


Figure 28 - CTS9513 Counter Mode X Representative Waveforms

MODE X HARDWARE SAVE

Mode X is a hardware edge triggered strobe counter with the capability of reading the counter value without interrupting the count.

As shown in Figure 28, once the counter is ARMed a valid gate edge starts the counter. Once triggered the counter will count to TC regardless of the state of the Gate line. Gate edges received prior to TC will store the current count in the Hold register. Once the counter has reached TC the counter will stop until a subsequent gate edge is received. Gate edges applied to an unarmed counter have no effect.

Symbol	**Specification**	**Min**	**Max**	**Units**
V_{ILT}	TTL Input LOW Level		0.8	Volts
V_{IHT}	TTL Input HIGH Level	2		Volts
V_{ILC}	X2 Input LOW Level		1.5	Volts
V_{IHC}	X2 Input HIGH Level	V_{DD} –1.5		Volts
V_{OL}	Output LOW Level @ I_{OL} = 4mA		0.4	Volts
V_{OH}	Output HIGH Level @ I_{OL} =4mA	2.4		Volts
I_{Z}	Input Leakage Current	-10	10	μA
I_{DD}	Supply Current /No Load / F_{OSC} = 7MHz		20	mA
I_{DDS}	IDD Static		10	μA
C_{IN}	Pin Capacitance		10	pF

Table 12 - CTS9513 Electrical Characteristics

Intel 82C55A Programmable Peripheral Interface
Data Sheet Reprint

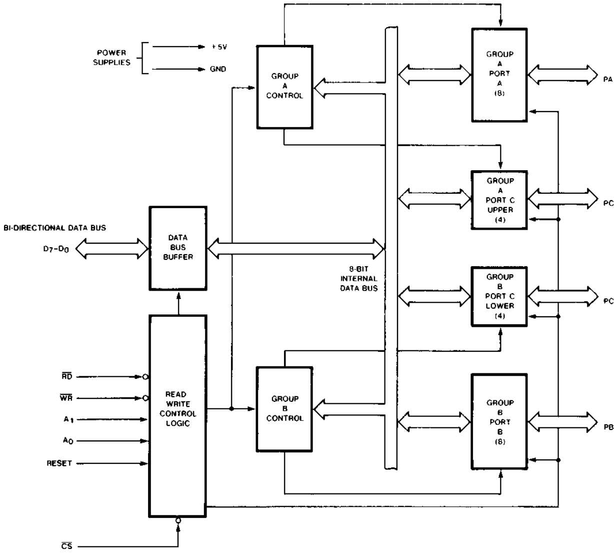
82C55A
CHMOS PROGRAMMABLE PERIPHERAL INTERFACE

- Compatible with all Intel and Most Other Microprocessors
- High Speed, "Zero Wait State" Operation with 8 MHz 8086/88 and 80186/188
- 24 Programmable I/O Pins
- Low Power CHMOS
- Completely TTL Compatible
- Control Word Read-Back Capability
- Direct Bit Set/Reset Capability
- 2.5 mA DC Drive Capability on all I/O Port Outputs
- Available in 40-Pin DIP and 44-Pin PLCC
- Available in EXPRESS
  — Standard Temperature Range
  — Extended Temperature Range

The Intel 82C55A is a high-performance, CHMOS version of the industry standard 8255A general purpose programmable I/O device which is designed for use with all Intel and most other microprocessors. It provides 24 I/O pins which may be individually programmed in 2 groups of 12 and used in 3 major modes of operation. The 82C55A is pin compatible with the NMOS 8255A and 8255A-5.

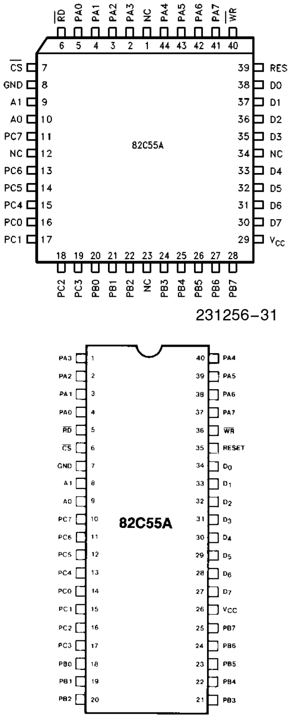
In MODE 0, each group of 12 I/O pins may be programmed in sets of 4 and 8 to be inputs or outputs. In MODE 1, each group may be programmed to have 8 lines of input or output. 3 of the remaining 4 pins are used for handshaking and interrupt control signals. MODE 2 is a strobed bi-directional bus configuration.

The 82C55A is fabricated on Intel's advanced CHMOS III technology which provides low power consumption with performance equal to or greater than the equivalent NMOS product. The 82C55A is available in 40-pin DIP and 44-pin plastic leaded chip carrier (PLCC) packages.



231256–1

Figure 1. 82C55A Block Diagram



231256–2

Figure 2. 82C55A Pinout

Diagrams are for pin reference only. Package sizes are not to scale.

Table 1. Pin Description

<table>
<tr><th rowspan="2">Symbol</th><th colspan="2">Pin Number</th><th rowspan="2">Type</th><th rowspan="2">Name and Function</th></tr>
<tr><th>Dip</th><th>PLCC</th></tr>
<tr><td>PA$_{3-0}$</td><td>1–4</td><td>2–5</td><td>I/O</td><td><b>PORT A, PINS 0–3:</b> Lower nibble of an 8-bit data output latch/buffer and an 8-bit data input latch.</td></tr>
<tr><td>$\overline{RD}$</td><td>5</td><td>6</td><td>I</td><td><b>READ CONTROL:</b> This input is low during CPU read operations.</td></tr>
<tr><td>$\overline{CS}$</td><td>6</td><td>7</td><td>I</td><td><b>CHIP SELECT:</b> A low on this input enables the 82C55A to respond to $\overline{RD}$ and $\overline{WR}$ signals. $\overline{RD}$ and WR are ignored otherwise.</td></tr>
<tr><td>GND</td><td>7</td><td>8</td><td></td><td><b>System Ground</b></td></tr>
<tr><td>A$_{1-0}$</td><td>8–9</td><td>9–10</td><td>I</td><td><b>ADDRESS:</b> These input signals, in conjunction $\overline{RD}$ and $\overline{WR}$, control the selection of one of the three ports or the control word registers.
<table>
<tr><th>A$_1$</th><th>A$_0$</th><th>$\overline{RD}$</th><th>$\overline{WR}$</th><th>$\overline{CS}$</th><th>Input Operation (Read)</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>Port A - Data Bus</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>0</td><td>Port B - Data Bus</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>Port C - Data Bus</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>Control Word - Data Bus</td></tr>
<tr><th colspan="6">Output Operation (Write)</th></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>Data Bus - Port A</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>0</td><td>Data Bus - Port B</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>Data Bus - Port C</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>Data Bus - Control</td></tr>
<tr><th colspan="6">Disable Function</th></tr>
<tr><td>X</td><td>X</td><td>X</td><td>X</td><td>1</td><td>Data Bus - 3 - State</td></tr>
<tr><td>X</td><td>X</td><td>1</td><td>1</td><td>0</td><td>Data Bus - 3 - State</td></tr>
</table></td></tr>
<tr><td>PC$_{7-4}$</td><td>10–13</td><td>11,13–15</td><td>I/O</td><td><b>PORT C, PINS 4–7:</b> Upper nibble of an 8-bit data output latch/buffer and an 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal outputs and status signal inputs in conjunction with ports A and B.</td></tr>
<tr><td>PC$_{0-3}$</td><td>14–17</td><td>16–19</td><td>I/O</td><td><b>PORT C, PINS 0–3:</b> Lower nibble of Port C.</td></tr>
<tr><td>PB$_{0-7}$</td><td>18–25</td><td>20–22,
24–28</td><td>I/O</td><td><b>PORT B, PINS 0–7:</b> An 8-bit data output latch/buffer and an 8-bit data input buffer.</td></tr>
<tr><td>V$_{CC}$</td><td>26</td><td>29</td><td></td><td><b>SYSTEM POWER:</b> + 5V Power Supply.</td></tr>
<tr><td>D$_{7-0}$</td><td>27–34</td><td>30–33,
35–38</td><td>I/O</td><td><b>DATA BUS:</b> Bi-directional, tri-state data bus lines, connected to system data bus.</td></tr>
<tr><td>RESET</td><td>35</td><td>39</td><td>I</td><td><b>RESET:</b> A high on this input clears the control register and all ports are set to the input mode.</td></tr>
<tr><td>$\overline{WR}$</td><td>36</td><td>40</td><td>I</td><td><b>WRITE CONTROL:</b> This input is low during CPU write operations.</td></tr>
<tr><td>PA$_{7-4}$</td><td>37–40</td><td>41–44</td><td>I/O</td><td><b>PORT A, PINS 4–7:</b> Upper nibble of an 8-bit data output latch/buffer and an 8-bit data input latch.</td></tr>
<tr><td>NC</td><td></td><td>1, 12,
23, 34</td><td></td><td>No Connect</td></tr>
</table>

82C55A FUNCTIONAL DESCRIPTION

General

The 82C55A is a programmable peripheral interface device designed for use in Intel microcomputer systems. Its function is that of a general purpose I/O component to interface peripheral equipment to the microcomputer system bus. The functional configuration of the 82C55A is programmed by the system software so that normally no external logic is necessary to interface peripheral devices or structures.

Data Bus Buffer

This 3-state bidirectional 8-bit buffer is used to interface the 82C55A to the system data bus. Data is transmitted or received by the buffer upon execution of input or output instructions by the CPU. Control words and status information are also transferred through the data bus buffer.

Read/Write and Control Logic

The function of this block is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the CPU Address and Control busses and in turn, issues commands to both of the Control Groups.

Group A and Group B Controls

The functional configuration of each port is programmed by the systems software. In essence, the CPU "outputs" a control word to the 82C55A. The control word contains information such as "mode", "bit set", "bit reset", etc., that initializes the functional configuration of the 82C55A.

Each of the Control blocks (Group A and Group B) accepts "commands" from the Read/Write Control Logic, receives "control words" from the internal data bus and issues the proper commands to its associated ports.

Control Group A - Port A and Port C upper (C7–C4)
Control Group B - Port B and Port C lower (C3–C0)

The control word register can be both written and read as shown in the address decode table in the pin descriptions. Figure 6 shows the control word format for both Read and Write operations. When the control word is read, bit D7 will always be a logic "1", as this implies control word mode information.

Ports A, B, and C

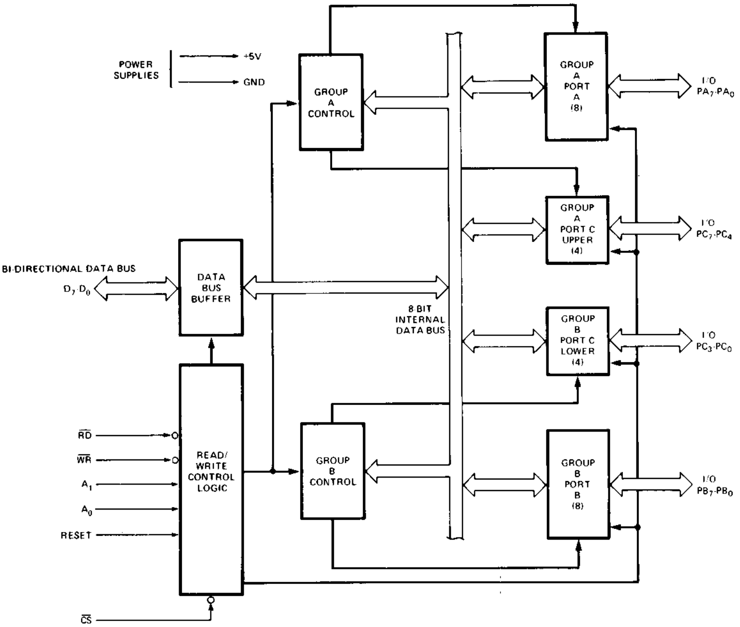
The 82C55A contains three 8-bit ports (A, B, and C). All can be configured in a wide variety of functional characteristics by the system software but each has its own special features or "personality" to further enhance the power and flexibility of the 82C55A.

Port A. One 8-bit data output latch/buffer and one 8-bit input latch buffer. Both "pull-up" and "pull-down" bus hold devices are present on Port A.

Port B. One 8-bit data input/output latch/buffer. Only "pull-up" bus hold devices are present on Port B.

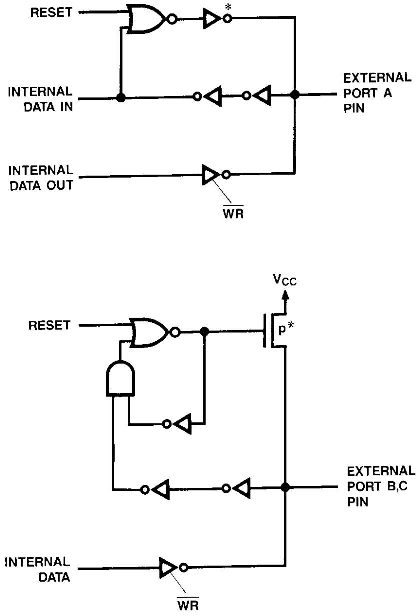
Port C. One 8-bit data output latch/buffer and one 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal outputs and status signal inputs in conjunction with ports A and B. Only "pull-up" bus hold devices are present on Port C.

See Figure 4 for the bus-hold circuit configuration for Port A, B, and C.



231256–3

Figure 3. 82C55A Block Diagram Showing Data Bus Buffer and Read/Write Control Logic Functions



231256–4

***NOTE:**
Port pins loaded with more than 20 pF capacitance may not have their logic level guaranteed following a hardware reset.

Figure 4. Port A, B, C, Bus-hold Configuration

82C55A OPERATIONAL DESCRIPTION

Mode Selection

There are three basic modes of operation that can be selected by the system software:

Mode 0 — Basic input/output
Mode 1 — Strobed Input/output
Mode 2 — Bi-directional Bus

When the reset input goes "high" all ports will be set to the input mode with all 24 port lines held at a logic "one" level by the internal bus hold devices (see Figure 4 Note). After the reset is removed the 82C55A can remain in the input mode with no additional initialization required. This eliminates the need for pullup or pulldown devices in "all CMOS" designs. During the execution of the system program, any of the other modes may be selected by using a single output instruction. This allows a single 82C55A to service a variety of peripheral devices with a simple software maintenance routine.

The modes for Port A and Port B can be separately defined, while Port C is divided into two portions as required by the Port A and Port B definitions. All of the output registers, including the status flip-flops, will be reset whenever the mode is changed. Modes may be combined so that their functional definition can be "tailored" to almost any I/O structure. For instance; Group B can be programmed in Mode 0 to monitor simple switch closings or display computational results, Group A could be programmed in Mode 1 to monitor a keyboard or tape reader on an interrupt-driven basis.

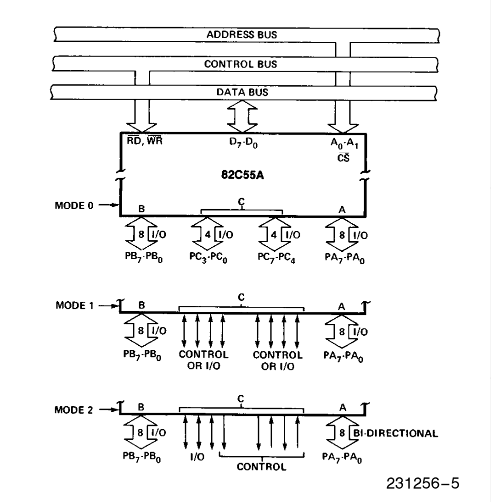


Figure 5. Basic Mode Definitions and Bus Interface

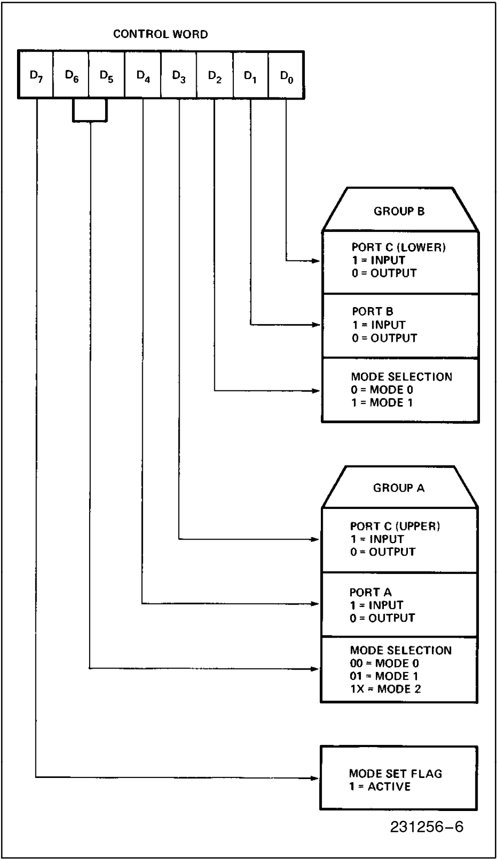


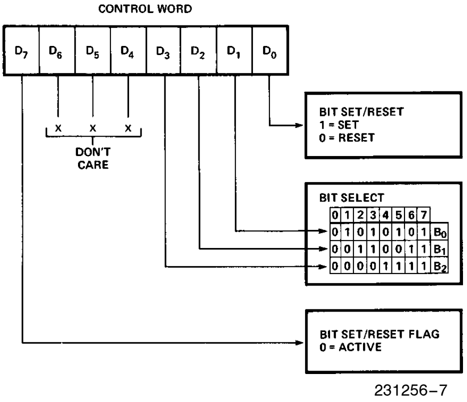
Figure 6. Mode Definition Format

The mode definitions and possible mode combinations may seem confusing at first but after a cursory review of the complete device operation a simple, logical I/O approach will surface. The design of the 82C55A has taken into account things such as efficient PC board layout, control signal definition vs PC layout and complete functional flexibility to support almost any peripheral device with no external logic. Such design represents the maximum use of the available pins.

Single Bit Set/Reset Feature

Any of the eight bits of Port C can be Set or Reset using a single OUTput instruction. This feature reduces software requirements in Control-based applications.

When Port C is being used as status/control for Port A or B, these bits can be set or reset by using the Bit Set/Reset operation just as if they were data output ports.



0	1	2	3	4	5	6	7	
0	1	0	1	0	1	0	1	B_0
0	0	1	1	0	0	1	1	B_1
0	0	0	0	1	1	1	1	B_2

231256–7

Figure 7. Bit Set/Reset Format

Interrupt Control Functions

When the 82C55A is programmed to operate in mode 1 or mode 2, control signals are provided that can be used as interrupt request inputs to the CPU. The interrupt request signals, generated from port C, can be inhibited or enabled by setting or resetting the associated INTE flip-flop, using the bit set/reset function of port C.

This function allows the Programmer to disallow or allow a specific I/O device to interrupt the CPU without affecting any other device in the interrupt structure.

INTE flip-flop definition:

(BIT-SET)—INTE is SET—Interrupt enable
(BIT-RESET)—INTE is RESET—Interrupt disable

Note:
All Mask flip-flops are automatically reset during mode selection and device Reset.

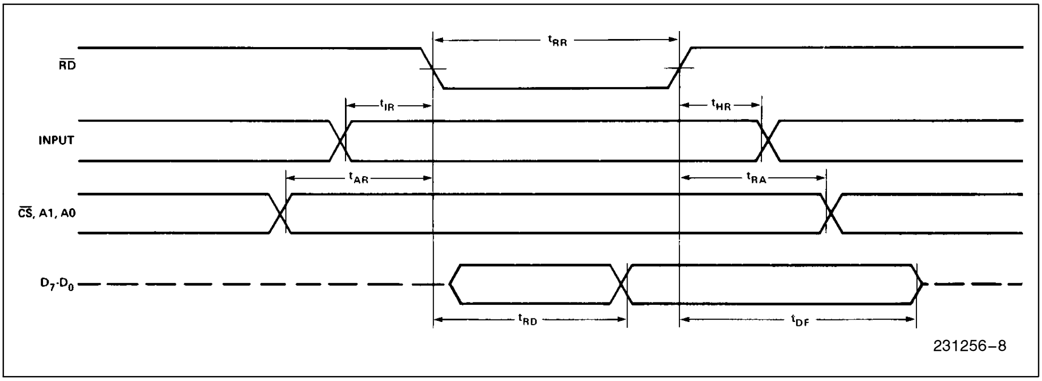
Operating Modes

Mode 0 (Basic Input/Output). This functional configuration provides simple input and output operations for each of the three ports. No "handshaking" is required, data is simply written to or read from a specified port.

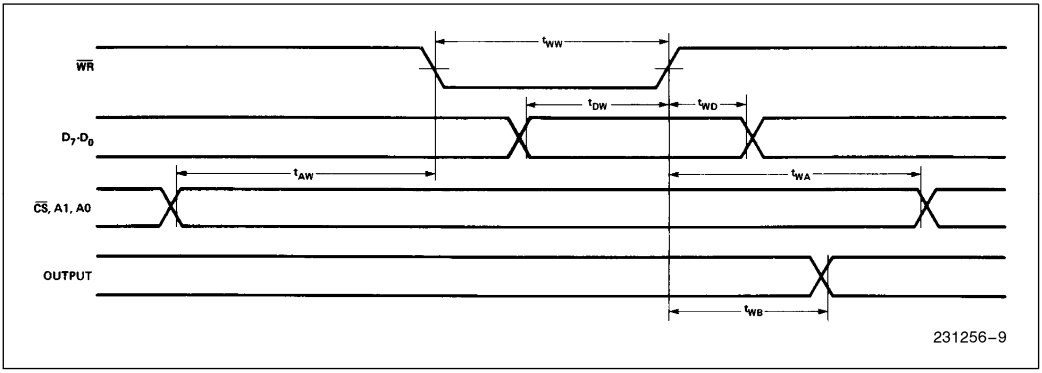
Mode 0 Basic Functional Definitions:

- Two 8-bit ports and two 4-bit ports.
- Any port can be input or output.
- Outputs are latched.
- Inputs are not latched.
- 16 different Input/Output configurations are possible in this Mode.

MODE 0 (BASIC INPUT)



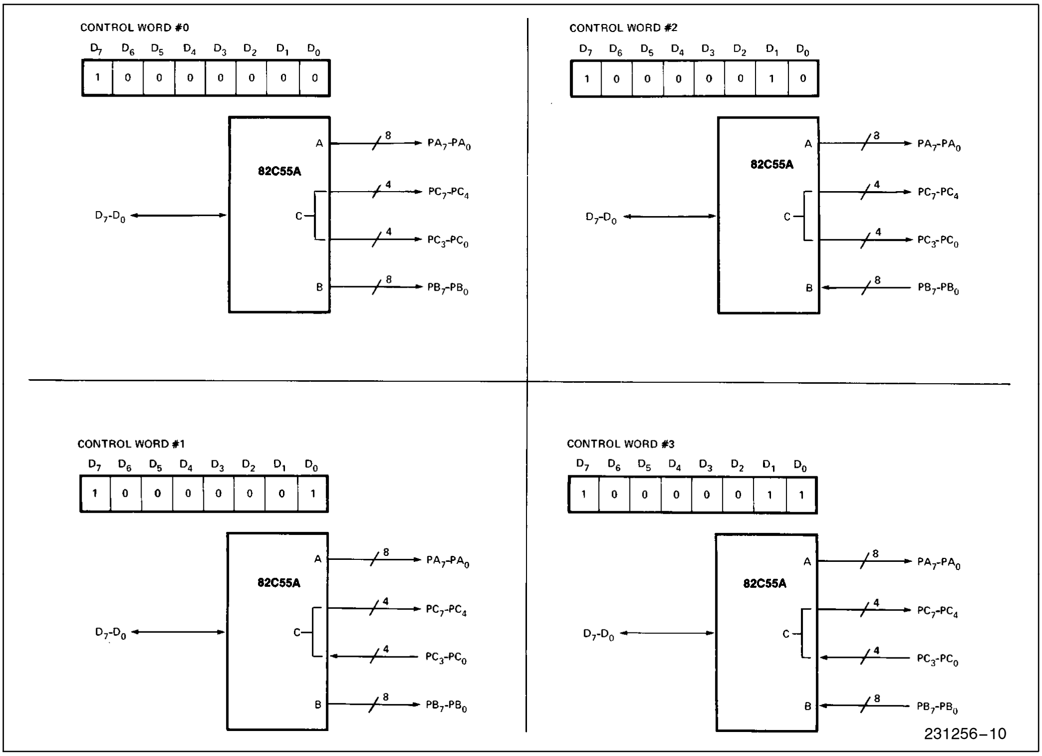
MODE 0 (BASIC OUTPUT)



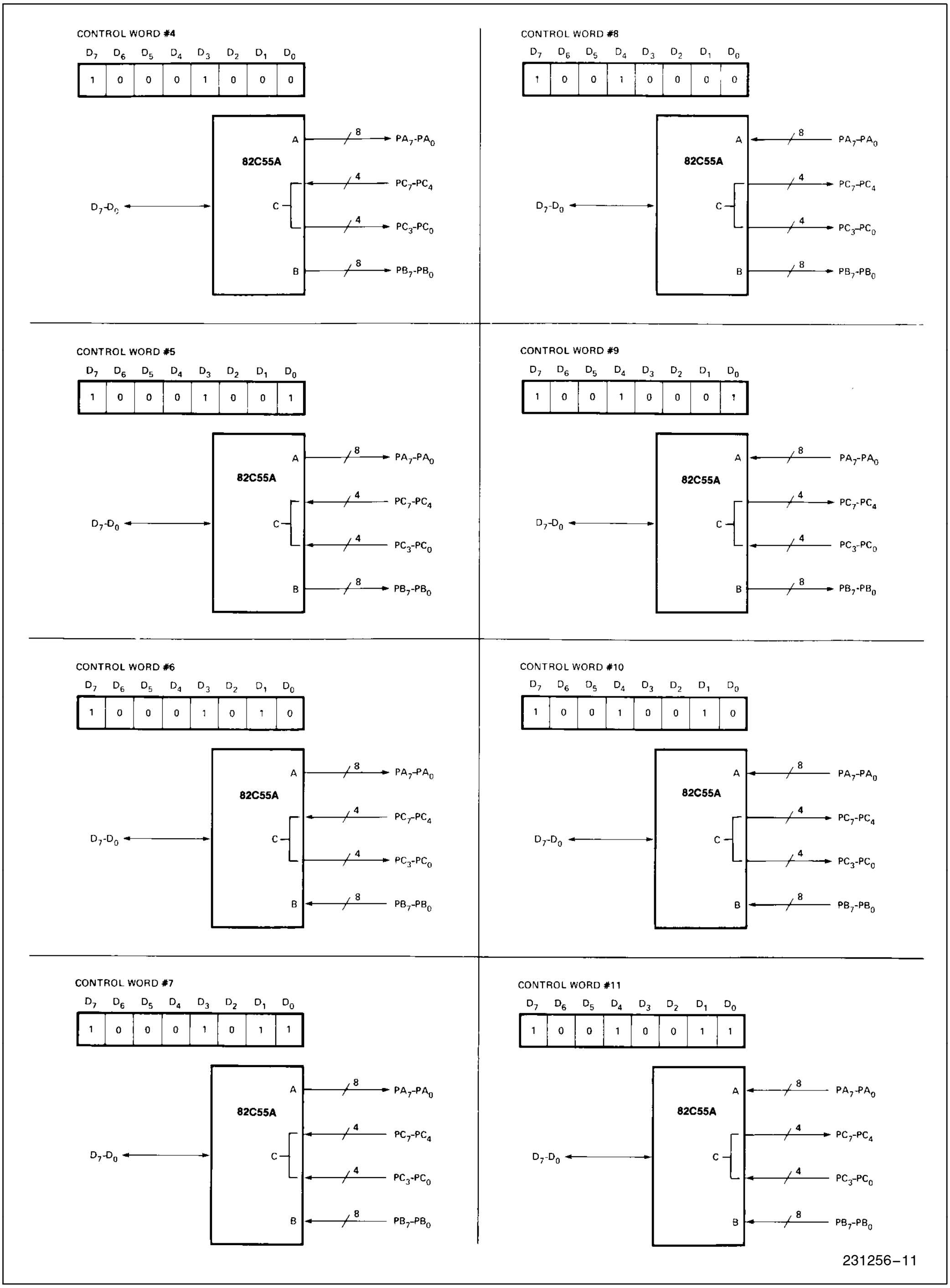
MODE 0 Port Definition

A		B		GROUP A			GROUP B	
D_4	D_3	D_1	D_0	PORT A	PORT C (UPPER)	#	PORT B	PORT C (LOWER)
0	0	0	0	OUTPUT	OUTPUT	0	OUTPUT	OUTPUT
0	0	0	1	OUTPUT	OUTPUT	1	OUTPUT	INPUT
0	0	1	0	OUTPUT	OUTPUT	2	INPUT	OUTPUT
0	0	1	1	OUTPUT	OUTPUT	3	INPUT	INPUT
0	1	0	0	OUTPUT	INPUT	4	OUTPUT	OUTPUT
0	1	0	1	OUTPUT	INPUT	5	OUTPUT	INPUT
0	1	1	0	OUTPUT	INPUT	6	INPUT	OUTPUT
0	1	1	1	OUTPUT	INPUT	7	INPUT	INPUT
1	0	0	0	INPUT	OUTPUT	8	OUTPUT	OUTPUT
1	0	0	1	INPUT	OUTPUT	9	OUTPUT	INPUT
1	0	1	0	INPUT	OUTPUT	10	INPUT	OUTPUT
1	0	1	1	INPUT	OUTPUT	11	INPUT	INPUT
1	1	0	0	INPUT	INPUT	12	OUTPUT	OUTPUT
1	1	0	1	INPUT	INPUT	13	OUTPUT	INPUT
1	1	1	0	INPUT	INPUT	14	INPUT	OUTPUT
1	1	1	1	INPUT	INPUT	15	INPUT	INPUT

MODE 0 Configurations



MODE 0 Configurations (Continued)



CONTROL WORD #4

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	1	0	0	0

CONTROL WORD #8

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	1	0	0	0	0

CONTROL WORD #5

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	1	0	0	1

CONTROL WORD #9

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	1	0	0	0	1

CONTROL WORD #6

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	1	0	1	0

CONTROL WORD #10

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	1	0	0	1	0

CONTROL WORD #7

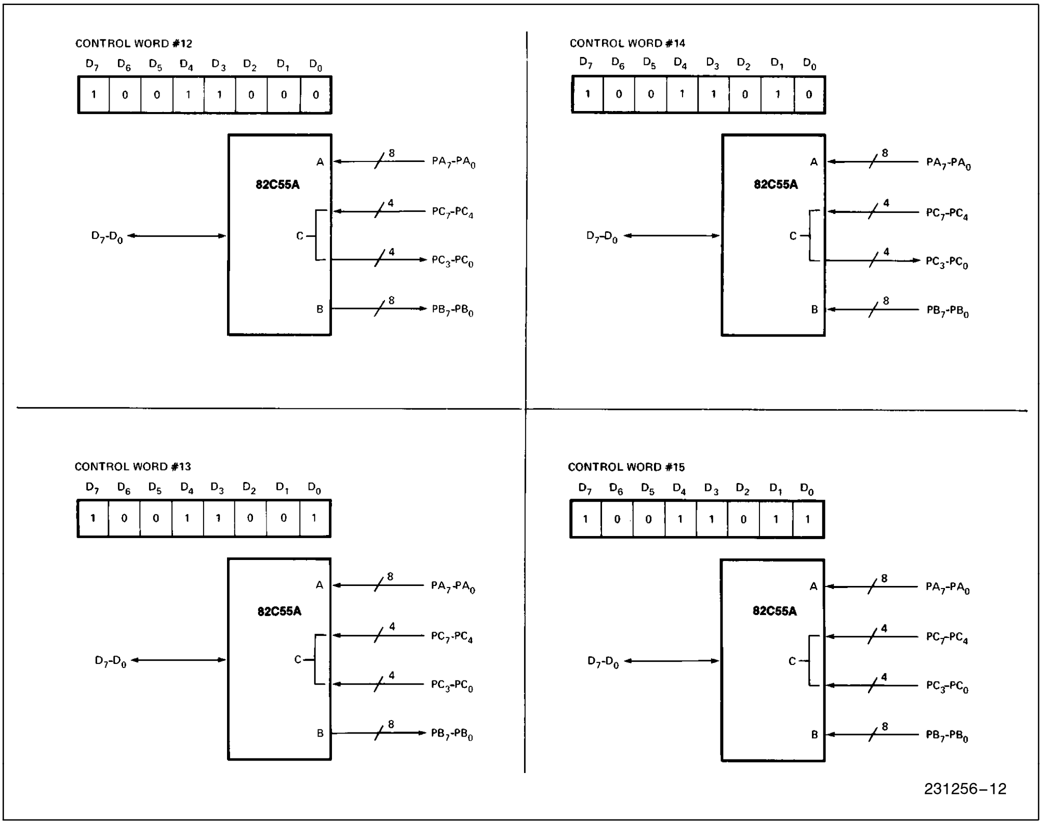
D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	1	0	1	1

CONTROL WORD #11

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	1	0	0	1	1

231256–11

MODE 0 Configurations (Continued)



231256–12

Operating Modes

MODE 1 (Strobed Input/Output). This functional configuration provides a means for transferring I/O data to or from a specified port in conjunction with strobes or "handshaking" signals. In mode 1, Port A and Port B use the lines on Port C to generate or accept these "handshaking" signals.

Mode 1 Basic functional Definitions:

- Two Groups (Group A and Group B).
- Each group contains one 8-bit data port and one 4-bit control/data port.
- The 8-bit data port can be either input or output Both inputs and outputs are latched.
- The 4-bit port is used for control and status of the 8-bit data port.

Input Control Signal Definition

$\overline{STB}$ (Strobe Input). A "low" on this input loads data into the input latch.

IBF (Input Buffer Full F/F)

A "high" on this output indicates that the data has been loaded into the input latch; in essence, an acknowledgement. IBF is set by $\overline{STB}$ input being low and is reset by the rising edge of the $\overline{RD}$ input.

INTR (Interrupt Request)

A "high" on this output can be used to interrupt the CPU when an input device is requesting service. INTR is set by the $\overline{STB}$ is a "one", IBF is a "one" and INTE is a "one". It is reset by the falling edge of $\overline{RD}$. This procedure allows an input device to request service from the CPU by simply strobing its data into the port.

INTE A

Controlled by bit set/reset of PC_4.

INTE B

Controlled by bit set/reset of PC_2.

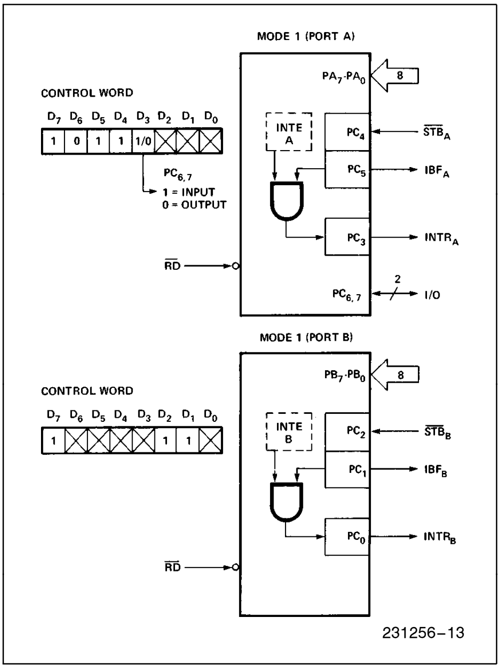


Figure 8. MODE 1 Input

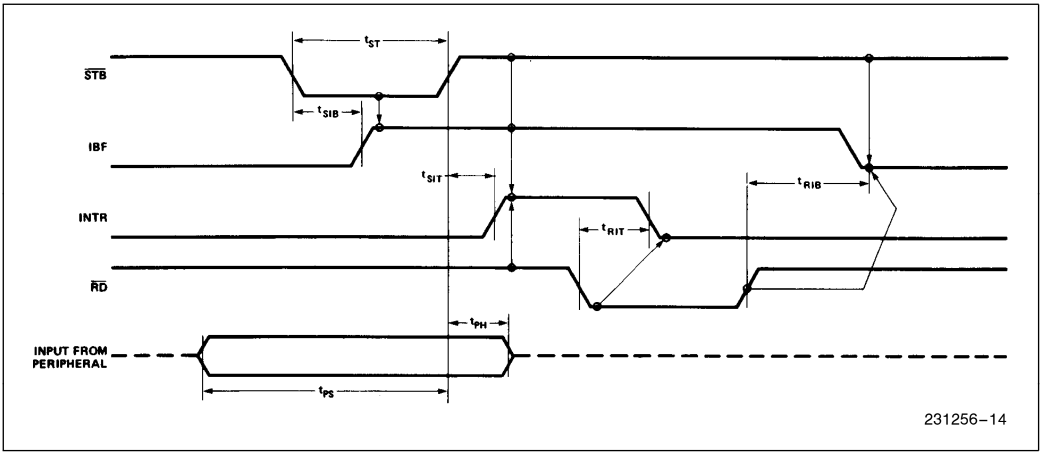


Figure 9. MODE 1 (Strobed Input)

Output Control Signal Definition

$\overline{OBF}$ (Output Buffer Full F/F). The $\overline{OBF}$ output will go "low" to indicate that the CPU has written data out to the specified port. The $\overline{OBF}$ F/F will be set by the rising edge of the $\overline{WR}$ input and reset by $\overline{ACK}$ Input being low.

$\overline{ACK}$ (Acknowledge Input). A "low" on this input informs the 82C55A that the data from Port A or Port B has been accepted. In essence, a response from the peripheral device indicating that it has received the data output by the CPU.

INTR (Interrupt Request). A "high" on this output can be used to interrupt the CPU when an output device has accepted data transmitted by the CPU. INTR is set when $\overline{ACK}$ is a "one", $\overline{OBF}$ is a "one" and INTE is a "one". It is reset by the falling edge of $\overline{WR}$.

INTE A

Controlled by bit set/reset of PC_6.

INTE B

Controlled by bit set/reset of PC_2.

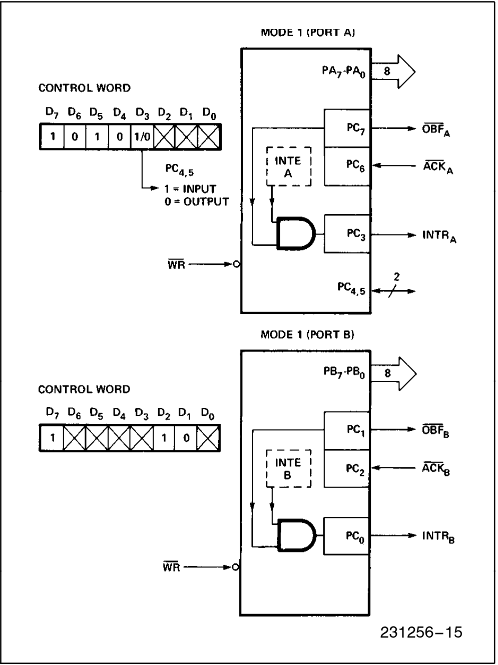


Figure 10. MODE 1 Output

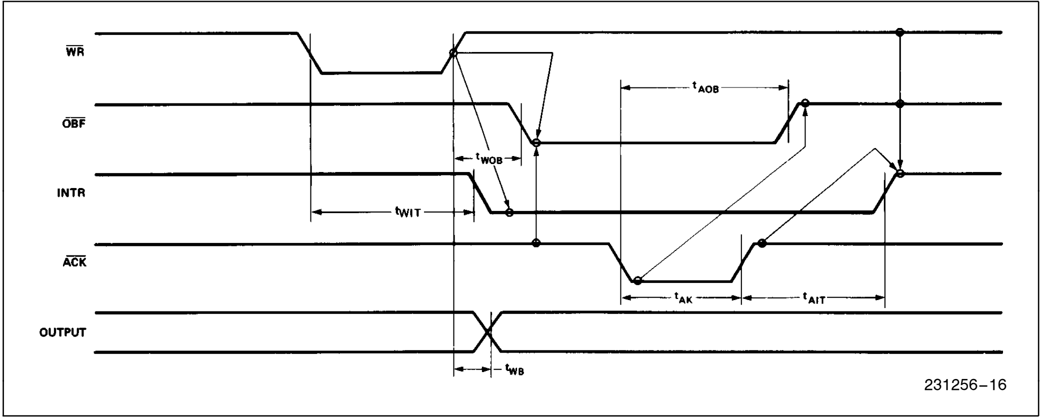


Figure 11. MODE 1 (Strobed Output)

Combinations of MODE 1

Port A and Port B can be individually defined as input or output in Mode 1 to support a wide variety of strobed I/O applications.

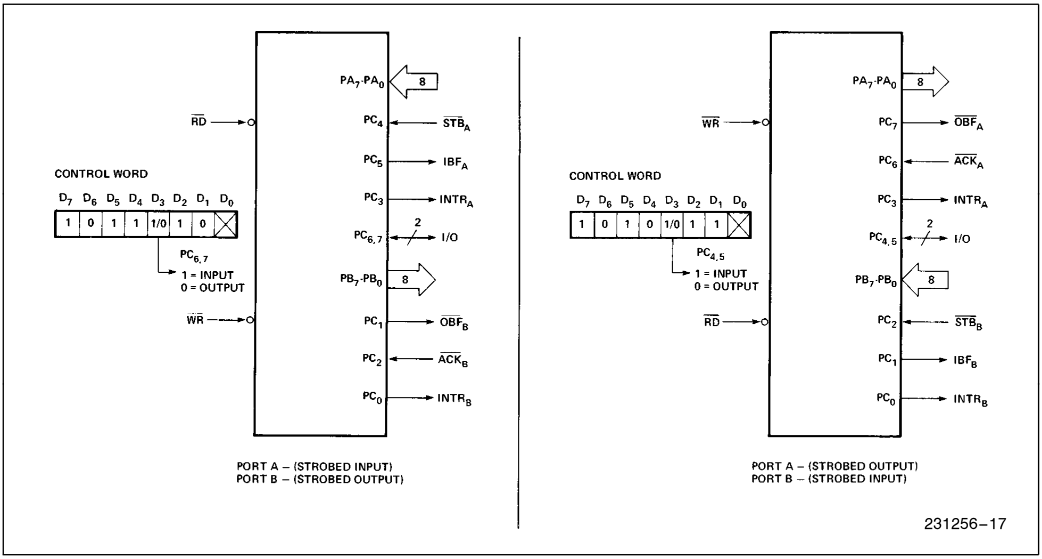


Figure 12. Combinations of MODE 1

Operating Modes

MODE 2 (Strobed Bidirectional Bus I/O). This functional configuration provides a means for communicating with a peripheral device or structure on a single 8-bit bus for both transmitting and receiving data (bidirectional bus I/O). "Handshaking" signals are provided to maintain proper bus flow discipline in a similar manner to MODE 1. Interrupt generation and enable/disable functions are also available.

MODE 2 Basic Functional Definitions:

- Used in Group A **only**.
- One 8-bit, bi-directional bus port (Port A) and a 5-bit control port (Port C).
- Both inputs and outputs are latched.
- The 5-bit control port (Port C) is used for control and status for the 8-bit, bi-directional bus port (Port A).

Bidirectional Bus I/O Control Signal Definition

INTR (Interrupt Request). A high on this output can be used to interrupt the CPU for input or output operations.

Output Operations

$\overline{\text{OBF}}$ (Output Buffer Full). The $\overline{\text{OBF}}$ output will go "low" to indicate that the CPU has written data out to port A.

$\overline{\text{ACK}}$ (Acknowledge). A "low" on this input enables the tri-state output buffer of Port A to send out the data. Otherwise, the output buffer will be in the high impedance state.

INTE 1 (The INTE Flip-Flop Associated with $\overline{\text{OBF}}$). Controlled by bit set/reset of PC_6.

Input Operations

$\overline{\text{STB}}$ (Strobe Input). A "low" on this input loads data into the input latch.

IBF (Input Buffer Full F/F). A "high" on this output indicates that data has been loaded into the input latch.

INTE 2 (The INTE Flip-Flop Associated with IBF). Controlled by bit set/reset of PC_4.

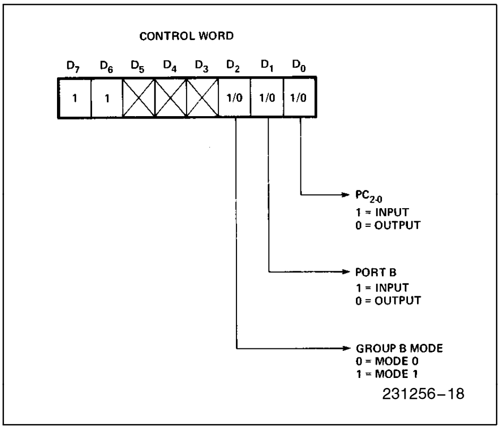


Figure 13. MODE Control Word

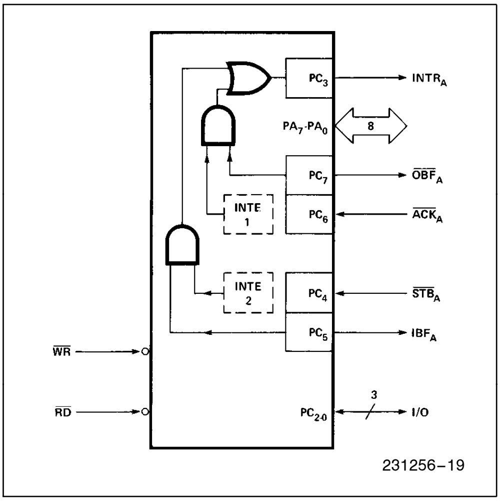


Figure 14. MODE 2

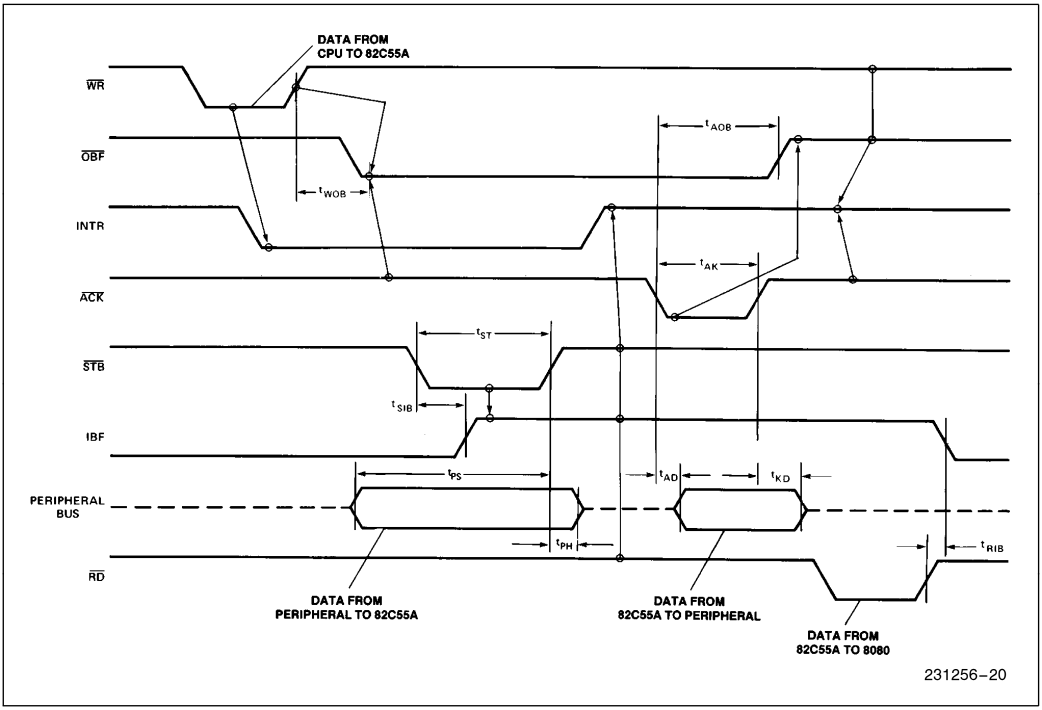


Figure 15. MODE 2 (Bidirectional)

NOTE:
Any sequence where $\overline{WR}$ occurs before $\overline{ACK}$, and $\overline{STB}$ occurs before $\overline{RD}$ is permissible.
$(INTR = IBF \bullet \overline{MASK} \bullet \overline{STB} \bullet \overline{RD} + \overline{OBF} \bullet \overline{MASK} \bullet \overline{ACK} \bullet \overline{WR})$

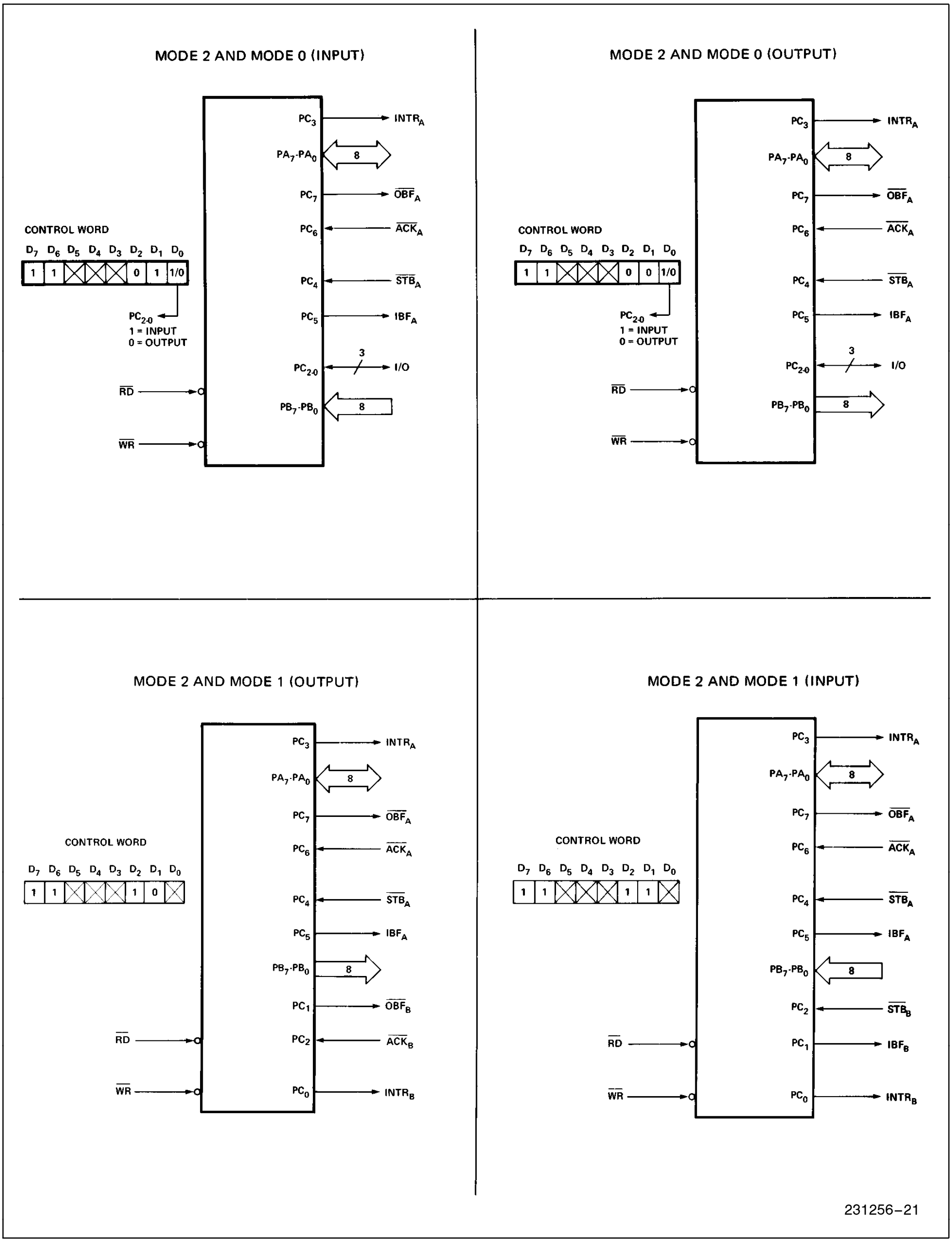


Figure 16. MODE ¼ Combinations

Mode Definition Summary

	MODE 0		MODE 1		MODE 2	
	IN	OUT	IN	OUT	GROUP A ONLY	
PA_0	IN	OUT	IN	OUT	⟷	
PA_1	IN	OUT	IN	OUT	⟷	
PA_2	IN	OUT	IN	OUT	⟷	
PA_3	IN	OUT	IN	OUT	⟷	
PA_4	IN	OUT	IN	OUT	⟷	
PA_5	IN	OUT	IN	OUT	⟷	
PA_6	IN	OUT	IN	OUT	⟷	
PA_7	IN	OUT	IN	OUT	⟷	
PB_0	IN	OUT	IN	OUT	—	MODE 0 OR MODE 1 ONLY
PB_1	IN	OUT	IN	OUT	—	
PB_2	IN	OUT	IN	OUT	—	
PB_3	IN	OUT	IN	OUT	—	
PB_4	IN	OUT	IN	OUT	—	
PB_5	IN	OUT	IN	OUT	—	
PB_6	IN	OUT	IN	OUT	—	
PB_7	IN	OUT	IN	OUT	—	
PC_0	IN	OUT	$INTR_B$	$INTR_B$	I/O	
PC_1	IN	OUT	IBF_B	$\overline{OBF}_B$	I/O	
PC_2	IN	OUT	$\overline{STB}_B$	$\overline{ACK}_B$	I/O	
PC_3	IN	OUT	$INTR_A$	$INTR_A$	$INTR_A$	
PC_4	IN	OUT	$\overline{STB}_A$	I/O	$\overline{STB}_A$	
PC_5	IN	OUT	IBF_A	I/O	IBF_A	
PC_6	IN	OUT	I/O	$\overline{ACK}_A$	$\overline{ACK}_A$	
PC_7	IN	OUT	I/O	$\overline{OBF}_A$	$\overline{OBF}_A$	

Special Mode Combination Considerations

There are several combinations of modes possible. For any combination, some or all of the Port C lines are used for control or status. The remaining bits are either inputs or outputs as defined by a "Set Mode" command.

During a read of Port C, the state of all the Port C lines, except the $\overline{\text{ACK}}$ and $\overline{\text{STB}}$ lines, will be placed on the data bus. In place of the $\overline{\text{ACK}}$ and $\overline{\text{STB}}$ line states, flag status will appear on the data bus in the PC2, PC4, and PC6 bit positions as illustrated by Figure 18.

Through a "Write Port C" command, only the Port C pins programmed as outputs in a Mode 0 group can be written. No other pins can be affected by a "Write Port C" command, nor can the interrupt enable flags be accessed. To write to any Port C output programmed as an output in a Mode 1 group or to change an interrupt enable flag, the "Set/Reset Port C Bit" command must be used.

With a "Set/Reset Port C Bit" command, any Port C line programmed as an output (including INTR, IBF and $\overline{\text{OBF}}$) can be written, or an interrupt enable flag can be either set or reset. Port C lines programmed as inputs, including $\overline{\text{ACK}}$ and $\overline{\text{STB}}$ lines, associated with Port C are not affected by a "Set/Reset Port C Bit" command. Writing to the corresponding Port C bit positions of the $\overline{\text{ACK}}$ and $\overline{\text{STB}}$ lines with the "Set/Reset Port C Bit" command will affect the Group A and Group B interrupt enable flags, as illustrated in Figure 18.

Current Drive Capability

Any output on Port A, B or C can sink or source 2.5 mA. This feature allows the 82C55A to directly drive Darlington type drivers and high-voltage displays that require such sink or source current.

Reading Port C Status

In Mode 0, Port C transfers data to or from the peripheral device. When the 82C55A is programmed to function in Modes 1 or 2, Port C generates or accepts "hand-shaking" signals with the peripheral device. Reading the contents of Port C allows the programmer to test or verify the "status" of each peripheral device and change the program flow accordingly.

There is no special instruction to read the status information from Port C. A normal read operation of Port C is executed to perform this function.

INPUT CONFIGURATION

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
I/O	I/O	IBF_A	$INTE_A$	$INTR_A$	$INTE_B$	IBF_B	$INTR_B$
GROUP A					GROUP B		

OUTPUT CONFIGURATIONS

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
$\overline{OBF}_A$	$INTE_A$	I/O	I/O	$INTR_A$	$INTE_B$	$\overline{OBF}_B$	$INTR_B$
GROUP A					GROUP B		

Figure 17a. MODE 1 Status Word Format

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
$\overline{OBF}_A$	$INTE_1$	IBF_A	$INTE_2$	$INTR_A$			
GROUP A					GROUP B (Defined By Mode 0 or Mode 1 Selection)		

Figure 17b. MODE 2 Status Word Format

Interrupt Enable Flag	Position	Alternate Port C Pin Signal (Mode)
INTE B	PC2	$\overline{ACK}_B$ (Output Mode 1) or $\overline{STB}_B$ (Input Mode 1)
INTE A2	PC4	$\overline{STB}_A$ (Input Mode 1 or Mode 2)
INTE A1	PC6	$\overline{ACK}_A$ (Output Mode 1 or Mode 2

Figure 18. Interrupt Enable Flags in Modes 1 and 2

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias....0°C to + 70°C

Storage Temperature ....− 65°C to + 150°C

Supply Voltage ....− 0.5 to + 8.0V

Operating Voltage ....+ 4V to + 7V

Voltage on any Input..........GND − 2V to + 6.5V

Voltage on any Output ..GND − 0.5V to V_{CC} + 0.5V

Power Dissipation ....1 Watt

NOTICE: This is a production data sheet. The specifications are subject to change without notice.

**WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.*

D.C. CHARACTERISTICS

T_A = 0°C to 70°C, V_{CC} = +5V ±10%, GND = 0V (T_A = −40°C to +85°C for Extended Temperture)

Symbol	Parameter	Min	Max	Units	Test Conditions
V_{IL}	Input Low Voltage	−0.5	0.8	V	
V_{IH}	Input High Voltage	2.0	V_{CC}	V	
V_{OL}	Output Low Voltage		0.4	V	I_{OL} = 2.5 mA
V_{OH}	Output High Voltage	3.0 V_{CC} − 0.4		V V	I_{OH} = −2.5 mA I_{OH} = −100 μA
I_{IL}	Input Leakage Current		±1	μA	V_{IN} = V_{CC} to 0V (Note 1)
I_{OFL}	Output Float Leakage Current		±10	μA	V_{IN} = V_{CC} to 0V (Note 2)
I_{DAR}	Darlington Drive Current	±2.5	(Note 4)	mA	Ports A, B, C R_{ext} = 500Ω V_{ext} = 1.7V
I_{PHL}	Port Hold Low Leakage Current	+50	+300	μA	V_{OUT} = 1.0V Port A only
I_{PHH}	Port Hold High Leakage Current	−50	−300	μA	V_{OUT} = 3.0V Ports A, B, C
I_{PHLO}	Port Hold Low Overdrive Current	−350		μA	V_{OUT} = 0.8V
I_{PHHO}	Port Hold High Overdrive Current	+350		μA	V_{OUT} = 3.0V
I_{CC}	V_{CC} Supply Current		10	mA	(Note 3)
I_{CCSB}	V_{CC} Supply Current-Standby		10	μA	V_{CC} = 5.5V V_{IN} = V_{CC} or GND Port Conditions If I/P = Open/High O/P = Open Only With Data Bus = High/Low $\overline{CS}$ = High Reset = Low Pure Inputs = Low/High

NOTES:

1. Pins A_1, A_0, $\overline{CS}$, $\overline{WR}$, $\overline{RD}$, Reset.
2. Data Bus; Ports B, C.
3. Outputs open.
4. Limit output current to 4.0 mA.

CAPACITANCE

$T_A = 25°C, V_{CC} = GND = 0V$

Symbol	Parameter	Min	Max	Units	Test Conditions
C_{IN}	Input Capacitance		10	pF	Unmeasured plns returned to GND $f_c = 1$ MHz(5)
$C_{I/O}$	I/O Capacitance		20	pF	

NOTE:
5. Sampled not 100% tested.

A.C. CHARACTERISTICS

$T_A = 0°$ to $70°C$, $V_{CC} = +5V \pm 10\%$, $GND = 0V$

$T_A = -40°C$ to $+85°C$ for Extended Temperature

BUS PARAMETERS

READ CYCLE

Symbol	Parameter	82C55A-2		Units	Test Conditions
		Min	Max		
t_{AR}	Address Stable Before $\overline{RD}$ ↓	0		ns	
t_{RA}	Address Hold Time After $\overline{RD}$ ↑	0		ns	
t_{RR}	$\overline{RD}$ Pulse Width	150		ns	
t_{RD}	Data Delay from $\overline{RD}$ ↓		120	ns	
t_{DF}	$\overline{RD}$ ↑ to Data Floating	10	75	ns	
t_{RV}	Recovery Time between $\overline{RD}/\overline{WR}$	200		ns	

WRITE CYCLE

Symbol	Parameter	82C55A-2		Units	Test Conditions
		Min	Max		
t_{AW}	Address Stable Before $\overline{WR}$ ↓	0		ns	
t_{WA}	Address Hold Time After $\overline{WR}$ ↑	20		ns	Ports A & B
		20		ns	Port C
t_{WW}	$\overline{WR}$ Pulse Width	100		ns	
t_{DW}	Data Setup Time Before $\overline{WR}$ ↑	100		ns	
t_{WD}	Data Hold Time After $\overline{WR}$ ↑	30		ns	Ports A & B
		30		ns	Port C

OTHER TIMINGS

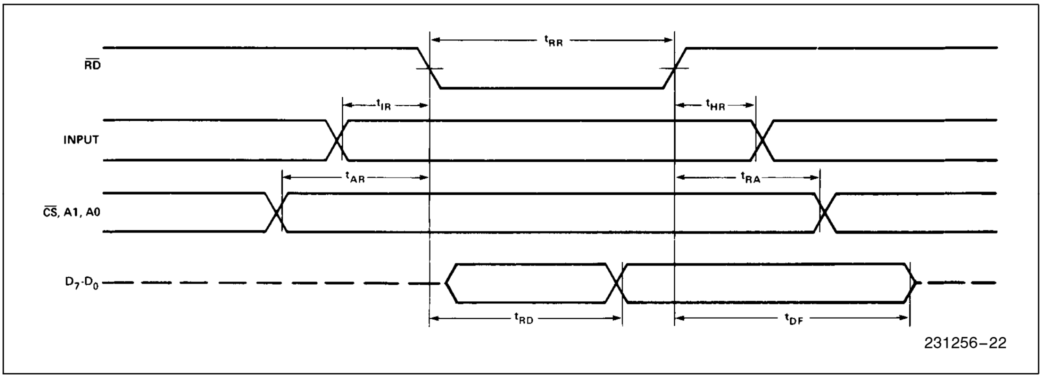
Symbol	Parameter	82C55A-2		Units Conditions	Test
		Min	Max		
t_{WB}	$\overline{WR}$ = 1 to Output		350	ns	
t_{IR}	Peripheral Data Before $\overline{RD}$	0		ns	
t_{HR}	Peripheral Data After $\overline{RD}$	0		ns	
t_{AK}	$\overline{ACK}$ Pulse Width	200		ns	
t_{ST}	$\overline{STB}$ Pulse Width	100		ns	
t_{PS}	Per. Data Before $\overline{STB}$ High	20		ns	
t_{PH}	Per. Data After $\overline{STB}$ High	50		ns	
t_{AD}	$\overline{ACK}$ = 0 to Output		175	ns	
t_{KD}	$\overline{ACK}$ = 1 to Output Float	20	250	ns	
t_{WOB}	$\overline{WR}$ = 1 to $\overline{OBF}$ = 0		150	ns	
t_{AOB}	$\overline{ACK}$ = 0 to $\overline{OBF}$ = 1		150	ns	
t_{SIB}	$\overline{STB}$ = 0 to IBF = 1		150	ns	
t_{RIB}	$\overline{RD}$ = 1 to IBF = 0		150	ns	
t_{RIT}	$\overline{RD}$ = 0 to INTR = 0		200	ns	
t_{SIT}	$\overline{STB}$ = 1 to INTR = 1		150	ns	
t_{AIT}	$\overline{ACK}$ = 1 to INTR = 1		150	ns	
t_{WIT}	$\overline{WR}$ = 0 to INTR = 0		200	ns	see note 1
t_{RES}	Reset Pulse Width	500		ns	see note 2

NOTE:

1. INTR ↑ may occur as early as $\overline{WR}$ ↓.
2. Pulse width of initial Reset pulse after power on must be at least 50 μSec. Subsequent Reset pulses may be 500 ns minimum. The output Ports A, B, or C may glitch low during the reset pulse but all port pins will be held at a logic "one" level after the reset pulse.

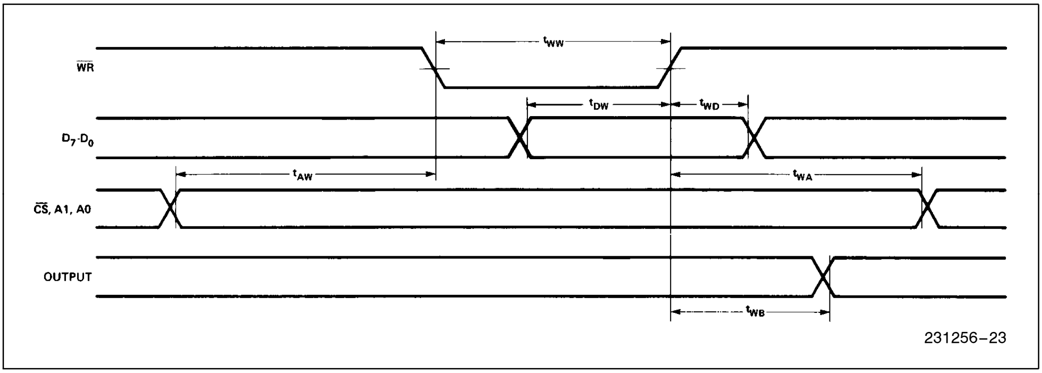
WAVEFORMS

MODE 0 (BASIC INPUT)



231256–22

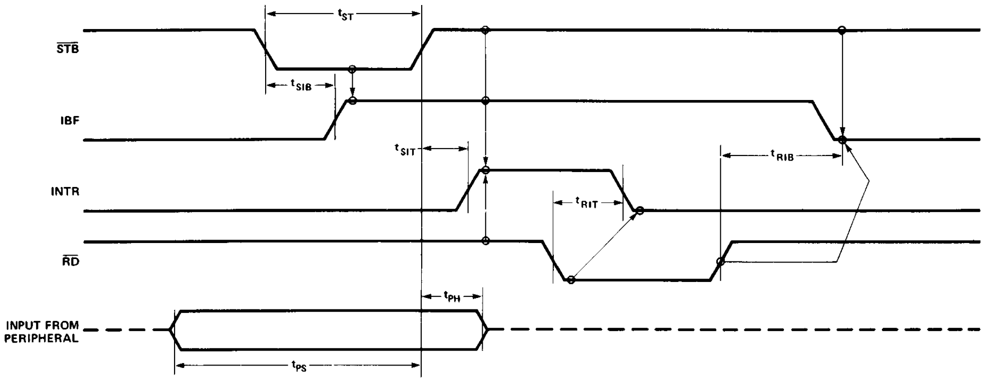
MODE 0 (BASIC OUTPUT)



231256–23

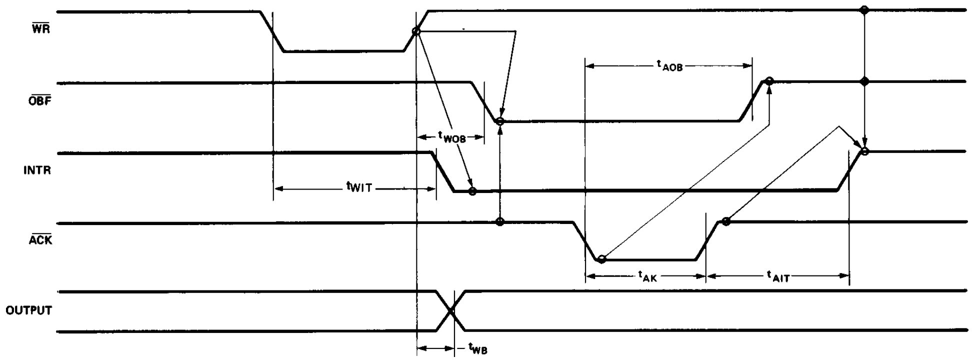
WAVEFORMS (Continued)

MODE 1 (STROBED INPUT)



231256–24

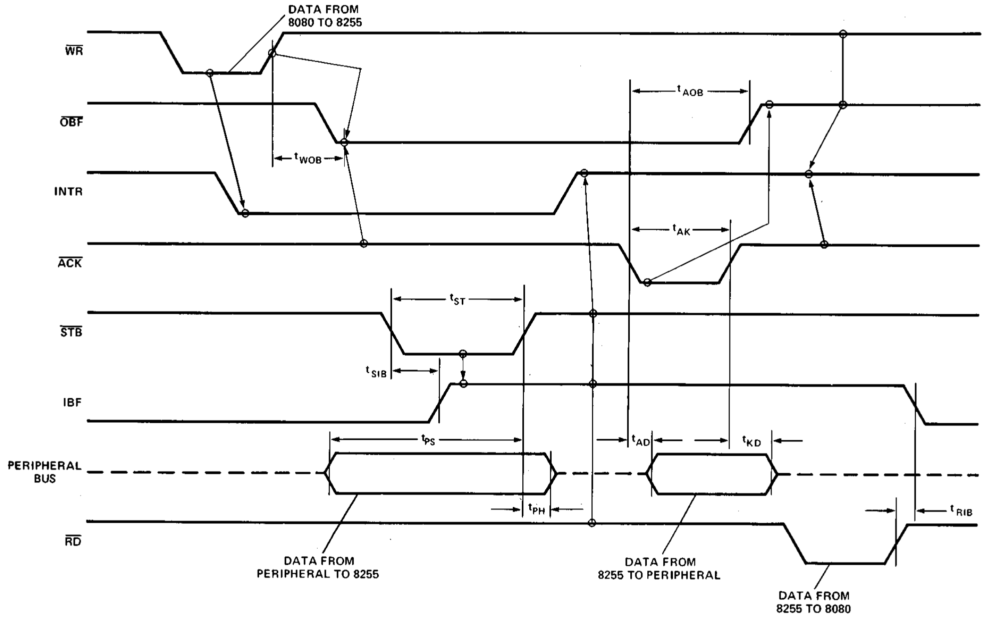
MODE 1 (STROBED OUTPUT)



231256–25

WAVEFORMS (Continued)

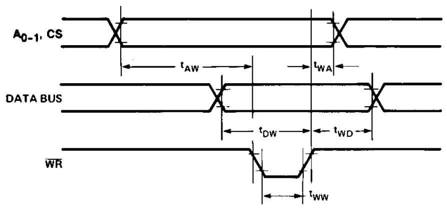
MODE 2 (BIDIRECTIONAL)



231256–26

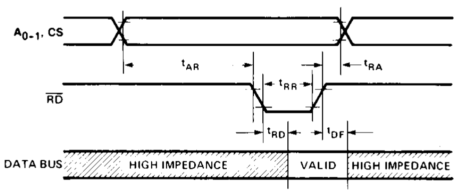
Note:
Any sequence where $\overline{WR}$ occurs before $\overline{ACK}$ AND $\overline{STB}$ occurs before $\overline{RD}$ is permissible.
(INTR = IBF • $\overline{MASK}$ • $\overline{STB}$ • $\overline{RD}$ + $\overline{OBF}$ • $\overline{MASK}$ • $\overline{ACK}$ • $\overline{WR}$)

WRITE TIMING



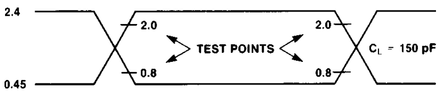
231256–27

READ TIMING



231256–28

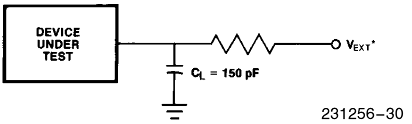
A.C. TESTING INPUT, OUTPUT WAVEFORM



231256–29

A.C. Testing Inputs Are Driven At 2.4V For A Logic 1 And 0.45V For A Logic 0 Timing Measurements Are Made At 2.0V For A Logic 1 And 0.8 For A Logic 0.

A.C. TESTING LOAD CIRCUIT



231256–30

*V_{EXT} Is Set At Various Voltages During Testing To Guarantee The Specification. C_L Includes Jig Capacitance.

APPENDIX D

WARRANTY AND RETURN POLICY

Return Policy

If you wish to return a product to the factory for service, please follow this procedure:

Read the Limited Warranty to familiarize yourself with our warranty policy.

Contact the factory for a Return Merchandise Authorization (RMA) number.

Please have the following available:

- Complete board name
- Board serial number
- A detailed description of the board's behavior

List the name of a contact person, familiar with technical details of the problem or situation, **along with their phone and fax numbers, address, and e-mail address** (if available).

List your shipping address!!

Indicate the shipping method you would like used to return the product to you.
We will not ship by next-day service without your pre-approval.

Carefully package the product, using proper anti-static packaging.

Write the RMA number in large (1") letters on the outside of the package.

Return the package to:

RTD Embedded Technologies, Inc.
103 Innovation Blvd.
State College PA 16803-0906
USA

LIMITED WARRANTY

RTD Embedded Technologies, Inc. warrants the hardware and software products it manufactures and produces to be free from defects in materials and workmanship for one year following the date of shipment from RTD Embedded Technologies, INC. This warranty is limited to the original purchaser of product and is not transferable.

During the one year warranty period, RTD Embedded Technologies will repair or replace, at its option, any defective products or parts at no additional charge, provided that the product is returned, shipping prepaid, to RTD Embedded Technologies. All replaced parts and products become the property of RTD Embedded Technologies. Before returning any product for repair, customers are required to contact the factory for an RMA number.

THIS LIMITED WARRANTY DOES NOT EXTEND TO ANY PRODUCTS WHICH HAVE BEEN DAMAGED AS A RESULT OF ACCIDENT, MISUSE, ABUSE (such as: use of incorrect input voltages, improper or insufficient ventilation, failure to follow the operating instructions that are provided by RTD Embedded Technologies, “acts of God” or other contingencies beyond the control of RTD Embedded Technologies), OR AS A RESULT OF SERVICE OR MODIFICATION BY ANYONE OTHER THAN RTD Embedded Technologies. EXCEPT AS EXPRESSLY SET FORTH ABOVE, NO OTHER WARRANTIES ARE EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, AND RTD Embedded Technologies EXPRESSLY DISCLAIMS ALL WARRANTIES NOT STATED HEREIN. ALL IMPLIED WARRANTIES, INCLUDING IMPLIED WARRANTIES FOR MECHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE LIMITED TO THE DURATION OF THIS WARRANTY. IN THE EVENT THE PRODUCT IS NOT FREE FROM DEFECTS AS WARRANTED ABOVE, THE PURCHASER’S SOLE REMEDY SHALL BE REPAIR OR REPLACEMENT AS PROVIDED ABOVE. UNDER NO CIRCUMSTANCES WILL RTD Embedded Technologies BE LIABLE TO THE PURCHASER OR ANY USER FOR ANY DAMAGES, INCLUDING ANY INCIDENTAL OR CONSEQUENTIAL DAMAGES, EXPENSES, LOST PROFITS, LOST SAVINGS, OR OTHER DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PRODUCT.

SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES FOR CONSUMER PRODUCTS, AND SOME STATES DO NOT ALLOW LIMITATIONS ON HOW LONG AN IMPLIED WARRANTY LASTS, SO THE ABOVE LIMITATIONS OR EXCLUSIONS MAY NOT APPLY TO YOU.

THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH VARY FROM STATE TO STATE.

RTD Embedded Technologies, Inc.
103 Innovation Blvd.
State College PA 16803-0906
USA
Our website: www.rtd.com

DM5804 Board User-Selected Settings

Base I/O Address:	
(hex)	(decimal)
IRQ Channel Selected (Select ONE Interrupt Source, P4):	
PC3	IRQ Channel #:
PC0	IRQ Channel #:
OUT5	IRQ Channel #:
EXTINT	IRQ Channel #: